Usage Guide for the STEP PDM Schema

August 13, 1999

Contacts

Jim Kindrick
ERIM/PDES, Inc.
3600 Green Court
Ann Arbor, MI 48105 USA
jkindrick@erim.org

Markus Hauser
ProSTEP GmbH
Julius-Reber-Str. 15
64293 Darmstadt, Germany
hauser@prostep.de

Release 3.0

Usage Guide for the STEP PDM Schema

Figures

Instance Diagrams

Exchange File Examples

Abstract

The STEP PDM Schema is a reference information model for the exchange of a central, common subset of the data being managed within a Product Data Management (PDM) system. It represents the intersection of requirements and data structures from a range of STEP Application Protocols (APs) all generally within the domains of design and development of discrete electromechanical parts and assemblies.

The STEP PDM Schema is *not* a specification for the functionality required for the complete scope of all PDM system functionality – i.e., it is *not* the union, but the intersection, of functionality present in the set of STEP Application Protocols. There exists functionality that is important for complete PDM functionality that is not represented in the PDM Schema, but in other units of functionality present in STEP APs.

By definition, a PDM system is something that manages data about products. At the central core of PDM information is product identification. A product in STEP represents the concept of a general managed item within a PDM system. In the STEP PDM Schema, the general product concept may be interpreted as either a Part (section 1) or a Document (section 4). In this way, parts and documents are managed in a consistent and parallel fashion.

Also central to the functionality of many PDM systems is identification of external files (both digital and physical), their relationship to managed documents, and their reference to core product identification. The external file reference mechanism in the STEP PDM Schema is described in section 5 of this document.

Classification of products is important in a PDM system for information classification and retrieval. It also supports basic type distinction between products that are parts and those that are documents. In the STEP PDM Schema, general product classification is used consistently for parts and documents.

Product properties are integrally related to the definition of an identified product, and so are naturally also included in the central core PDM information. Sections to be included in this document will discuss properties associated with an identified product, interpreted as either a part or a document.

Various general authorization and organizational data that are related to core product identification play an important role in PDM systems. Sections of this document to be included will describe the various organizational and management constructs that support product authorization in the STEP PDM Schema.

Product structures are the principle relationships that define assemblies and product configurations. Section 3 details part structures in the STEP PDM Schema; sections to be included will describe document structures. Configuration identification and effectivity information related to these structures will be detailed in sections to be included.

Finally, future sections will describe structures to manage the documentation of requests and corresponding orders for engineering action, in support of the change management process. Also included will be representations for contract and project identification.

Acknowledgements

Rogerio Barra
Markus Hauser
Oliver Holzel
Jim Kindrick
Achim Klein
Mario Leber
Andreas Trautheim
Max Ungerer
Glen Ziolko

Overview

Usage guide goal - To describe the recommended structure and attribute population for particular instance models created from the EXPRESS entities and types defined by the STEP PDM Schema. The selected instance models illustrate how to encode data values that need to be exchanged in support of key industry requirements common across the product manufacturing domain.

Usage guide status - This usage guide is a living document that will be extended over time to cover the entire scope of the PDM Schema. The maintenance of this document is guided by the PDM Implementor Forum where issues and clarifications are raised and resolved. Issues may be put forward to the points of contact identified on the cover page of this document.

Intended audience - Developers of applications and information management systems that must use product data and exchange it with other systems and applications in support of the business processes related to product design and development. Anyone interested in the scope of the requirements supported by the STEP PDM Schema.

Intended use - Manual and companion to the developer of STEP data exchange and translator software used by applications and information management systems that rely on product data. Guideline for consistent preprocessor instance model creation and requirement value encoding to enable meaningful information exchange between different systems and applications using STEP. Guideline for consistent interpretation by a postprocessor of a STEP Part 21 exchange file according to the unified STEP PDM Schema.

Usage guide style - Overall document proceedes in an incremental, step-by-step fashion to describe, and in parallel to illustrate, the recommended instantiation of the EXPRESS entities and types in the PDM Schema.

The diagram figures are presented using a graphical notation intended to illustrate the instance model. This notation *is not* EXPRESS-G and *does not* illustrate the EXPRESS schema; rather it is a graphical illustration of a specific population of a particular instance model of the schema. Furthermore, diagrams may omit attributes that are not deemed relevant in the context being discussed. This notation supports:

- illustration of entity instances,
- illustration and identification of referenced instances that are not fully illustrated in the current figure,
- illustration and identification of the multiple possible referenced instances corresponding to an attribute that has a select type as the value,
- indication of optional attributes and optional structure (dashed lines),
- illustration and identification of groups of functionally related instances (shaded bounding box),
- identification of specific attribute values (typically string values, maybe enumerated type values).

Each instance diagram figure is accompanied by a related STEP Part 21 exchange structure example. The example exchange file corresponds directly to the related instance model diagram and illustrates the very same thing using a different notation, i.e., STEP Part 21 syntax versus the graphical instance model notation.

Usage guide structure - The overall scope of requirements is partitioned into a set of major sections corresponding to identified units of functionality. Within a major section, there may be sub-sections. These sub-sections further divide the scope into smaller components of coherent functionality that interact with each other to realize the functionality of the entire unit.

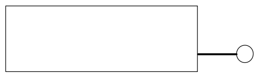
There is generally a description of requirements and a corresponding instance diagram associated with each section and sub-section of this document. Each instance diagram is followed by a detailed explanation and specific recommendations for the EXPRESS entities used in the instantiation diagram example. The entity listing and explanation is in turn followed by the corresponding STEP Part 21 exchange structure example.

Within a section, diagrams corresponding to sub-sections incrementally build upon one another to finally achieve a complete instance model example that illustrates the entire scope of the unit of functionality.

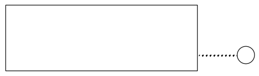
General Information

Instantiation diagrams - The diagrams are presented using a graphical notation intended to illustrate the instance model. This notation is not EXPRESS-G, however it does intentionally resemble it. The diagrams do not illustrate the EXPRESS schema, but illustrate a specific population of a particular instance model (or portion thereof) of the schema. A legend for the diagram notation is shown below :

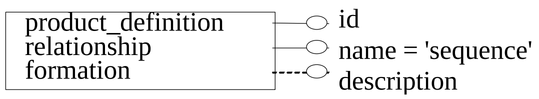
- illustration of entity instances and their attributes;



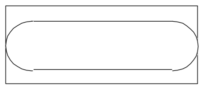
- identification of optional attributes;



- indication of specific attribute values, typically strings, may be enumerated values;



- identification of other document sections and referenced entity instances not fully illustrated in the current figure;



Entities and attributes not supported by the preprocessor - For various reasons, there may be some entities that cannot be completely exported by the preprocessor. Sometimes an application may not maintain all the information that is anticipated for the data exchange. Other times, the information may be maintained by a sending system but not included in the data exchange. Never the less, the preprocessor must provide values for all mandatory attributes in an exchange file. For mandatory string attribute values, the null (empty) string '' has often been used when a preprocessor can provide no real user data. The default string value '/NULL' may be used for this purpose, as recommended by the European automotive industry. When no data is provided by a sending system for a string value, the preprocessor should use '/NULL' or the empty string ''. To further indicate the reason why no data is provided, the following convention may be used:

- empty string '' indicates user data managed by the sending system but not provided for data exchange.
- string '/NULL' indicates user data in a mandatory attribute that is not managed by the sending system or currently not known.
- $ is used in the physical file, if an optional attribute is not instantiated.

It is generally not recommended to use the empty null string '' or the default string '/NULL' as valid user data.

Entities and attributes not supported by the postprocessor - For various reasons, there may be some entities that cannot be completely imported by the postprocessor. The postprocessor translator implementation simply may not support the import of the entity. The receiving system may not maintain the information that is carried by an entity or attribute, or it may require specific attribute values that are not present in the input data. Entities and attributes not imported should list a reason in a history log file. Entities and attributes not supported by the receiving system should not cause a system failure. The minimum acceptable behavior should be to ignore the unsupported constructs.

Unspecified and optional attribute values - Optional attributes without specific recommended values, such as the description attribute, are available on many entities in the PDM Schema. In general, use of this type of attribute is given the following recommendation:

***preprocessor* -** First, follow the usage guide as much as is possible **-** if some specific common harmonized user requirement has been documented in the usage guide for the attribute, try to adapt this requirement to those you have identified (i.e., map the standard into your user domain). If no specific common harmonized user requirement has been documented in the usage guide, in general such an optional attribute should not be instantiated. However, these attributes may be used in some bilateral agreements between exchange partners.

***postprocessor* -** Any optional attribute with no specific mapping specified in general can not be specifically interpreted in an interoperable way. While these types of attributes are in general not recommended to be instantiated, the postprocessor should gracefully handle any data that is exchanged using these attributes. A robust, interoperable PDM Schema processor will generally provide user access to the values exchanged.

Implementation project specific values - Attribute values recommended in this usage guide should be supported by systems conforming to the PDM Schema. Other values negotiated between exchange partners in specific projects may be used where the interpretation of their meaning does not contradict definitions provided in this usage guide. However, these agreements will not generally be interoperable solutions.

Leading and trailing blanks in STRING values - All white space within the single quote delimiters of a STRING value should be considered valid user data.

Uniqueness of identifiers - Identifiers can not be unique in general. In the parallel management of internal and external identifiers in a database, duplicate identifiers may occur that can cause problems with the uniqueness rules defined in the EXPRESS schema. In those cases, the interpretation of the identifier can be logically extended by a prefix that identifies the organizational scope (id value of the organization related in role 'id owner') to help ensure uniqueness (see1.1.1.1). The unique rule shall be evaluated only in scope of the organization defined by the id_owner. Thus, a processor may utilize the organization identifier as a key to identifying a product.

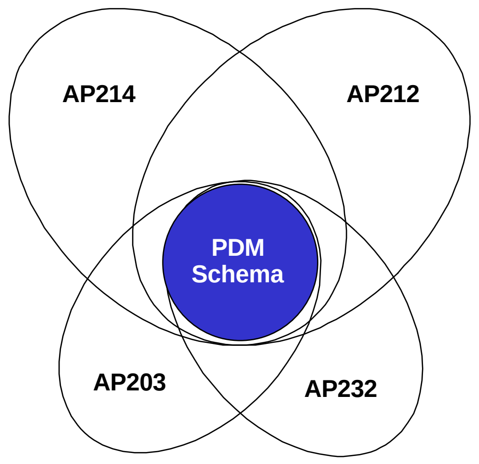
Schema version identification - Version identification for the PDM Schema shall be encoded in the header section of the STEP Part 21 exchange file to identify the version of the schema to which the file conforms. This is done with the header entity ***file_schema***, which identifies the EXPRESS schemas that specify the entity instances in the data section. The attribute ***schema_identifiers*** contains a list of strings that name the schema, optionally followed by the object identifier assigned to that schema. In place of the object identifier, the PDM Schema version identification number shall be enclosed within curly braces.
Only capital letters shall be used in schema name strings.

EXAMPLE : To indicate PDM Schema version 1.1, the following instance of the header entity file_schema should be used.

```
ISO-10303-21;
HEADER;
...
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
```

Scope of the STEP PDM Schema

The STEP PDM Schema is a core set of entities in STEP that support the mapping of concepts for Product Data Management (PDM). This document and the PDM Schema are the result of a cooperative development process between ProSTEP and PDES, Inc. The PDM Schema has been established to promote interoperability between STEP APs in the area of product data management.



☞ **Common PDM data schema generated and maintained by PDES, Inc., ProSTEP and JSTEP**

☞ **Real Subset of PDM relevant STEP APs (AP203, 212, 214, 232)**

☞ **Fulfills nearly all requirements for PDM data exchange**

Main functionality for parts and documents:
- identification
- versioning
- structures incl. transformations
- approvals and authorization
- project, work order, work request
- effectivities
- classification and properties

Figure 1: Positioning and Contents of the PDM Schema

This document summarizes implementation experiences of the PDM Schema version 1.1. It introduces a set of functional areas that together describe the scope of functionality of this release of the PDM Schema. These units of functionality suggest a modular structure for the PDM Schema.

Units of Functionality

The product data management requirements addressed by the PDM Schema are organized into groupings of related concepts. These groups provide a logical grouping of AIM entities for the purpose of a clear structure within the PDM Schema. These groups suggest a modular structure for the PDM Schema in line with the current direction towards modularization of the technical contents of STEP Integrated Resources and Application Protocols.

The modular semantic units of functionality are listed below:

- Part Identification,
- Part Classification,
- Part Properties,
- Part Structure and Relationships,

- Document Identification,
- Document Classification,
- Document and File Properties,
- Document Structure and Relationships,

- External Files,
- Document and file association to product data,

- Alias identification,
- Authorization,
- Configuration and Effectivity information,
- Work Management data.

1 Part Identification

The PDM Schema manages all parts as products, according to a fundamental STEP interpretation of 'Part as Product'. Part identification is achieved using basic product identification. The identified product may represent a part, a tool, or raw material. In addition, a product may represent a managed document and be identified according to the 'Document as Product' interpretation (see Section 4.1).

Part identification is the center for assignment of further product management data. As a consequence, at least one product identification (part or document) must be instantiated for the exchange of part/document data in the STEP PDM Schema.

The simple alias concept supports assignment of an alternate identifier to a product. This mechanism may also be used to alias other elements of product data.

Do not use the alias identification for the requirement of supplied part or document identification. This more complex alias relationship concept is specific for identification and renumbering of supplier's parts and documents.

1.1 Part as Product

The PDM Schema manages all parts as products. Identification of products in the STEP PDM Schema consists of three distinct concepts:

- Product Master Identification,
- Context Information,
- Type Classification.

Product master identification maintains the base part number, distinct part version identification, and information identifying a view definition. An identified part may be a single piece part or an assembly of arbitrary complexity.

Context information provides a scope and necessary circumstance for interpretation of product identification information.

Type classification distinguishes products representing parts from those that represent documents. Further possible type classifications like 'tool' or 'raw material' will be described in a future release in separate sections, e.g., to describe the relationship between a part and its raw material.

1.1.1 Product Master Identification

Product master identification supports the ability to uniquely identify a part. The identification of products in the STEP PDM Schema consists of three important and structurally distinct concepts:

- Base Identification,
- Version Identification,
- View Definition.

The master base identification maintains information common to all product versions, disciplines, and/or life-cycle views. It contains the base product number and name. The base number should not be subject to any encoding of information into a single complex parseable string.

There must be at least one version assigned to a product base identification. The version information may represent a design revision or iteration in the design cycle of a part. There may be more than one version associated with a product master. The set of versions associated with a base product may be related together to represent the version history of that product. The product version collects all information among all associated disciplines and life-cycle view definitions.

It is recommended that at least one view definition be assigned to each product version. There may be two exceptions to this general rule:

- Supplied product identification, in which case a supplied product may be represented by only product base;
- Version identification when version history is represented, where only the most recent current version is required to have an associated view definition.

The product definition view collects information relevant from the perspective of a particular application domain or life-cycle stage. There may be more than one life-cycle view definition associated with a particular version of a product. This is especially important to enable different views on product assembly structures.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of part master identification are illustrated in Diagram 1 below. The corresponding STEP exchange file encoding is given in Example 1.

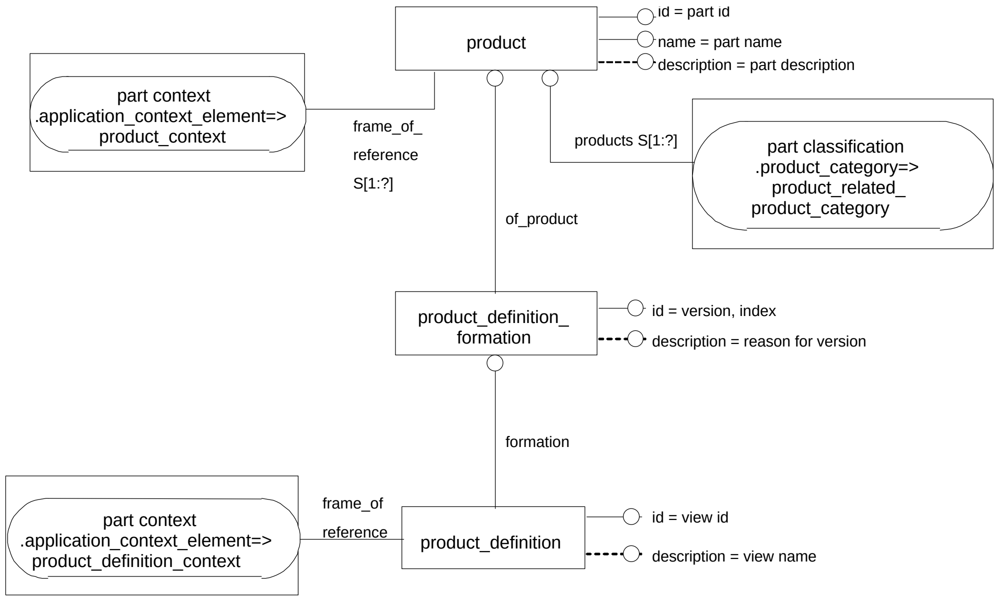


Diagram 1: Part Master Identification Instance Diagram

1.1.1.1 product

The product entity represents the product master base information. This entity collects all information that is common among the different versions and views of the product. The product number is strictly an identifier. It should not be used as a 'smart string' with some parseable internal coding scheme, e.g., to identify version or classification information.

The product number identifier must be unique within the scope of the business process of the information exchange. This is typically not a problem when the product data is only used within a single company. If the data is being assembled for an external use, the identification must be interpreted as unique within that broader domain. Processors may need to evaluate more than one string (i.e., product id) to establish unique identification of a part; there may be a combination of parameters that make part identification unique. The associated organization entity with the role 'id owner' can be used to derive a uniqueness parameter if the product.id attribute is not unique within the domain of the business arrangement of the exchange.

Attributes

- The ***id*** attribute stores the unique base identifier for a product, the product number.
- The ***name*** attribute stores the nomenclature, or common name, of the product.
- The ***description*** attribute should contain an expanded name or description of the product.
- Context information is stored in the ***frame_of_reference*** attribute. All products in STEP must be founded in some product_context. This requirement is discussed in more detail in 1.1.2.

ENTITY product	Attribute Population	Remarks
id	type: identifier = string product number identification	Unique within the scope provided by organization in role 'id owner'
name	type: label = string	
description	type: text = string	
frame_of_reference	type: entity = product_context	SET[1:?]

Preprocessor Recommendations: All preprocessors should use valid user (non-defaulted) data for the values assigned to the attributes id and name, as well as to the id of the organization related in the role ‘id owner’. In populating the id attribute, the identifier must be unique within the scope of the related organization or person_and_organization in the role 'id owner'.

Postprocessor Recommendations: Postprocessors should provide user access to the values of the attributes id and name, as well as the id of the organization related in the role 'id owner'.

Related Entities: The PDM Schema requires that all products exist in at least one product_category, to provide type classification information. This type classification requirement is discussed in 1.1.3.

It is strongly recommended that products have a related organization or person_and_organization that identifies responsibility for the original design of the part. The role of this assignment is identified as the ‘id owner’. This is the organization or person_and_organization that defines the part and assigns the part number. See 9.1 for guidance on how to create the organization or person_and_organization entities.

In addition, the product may optionally be referenced by the following entities:

- applied_organization_assignment (or applied_person_and_organization_assignment) to represent the customer, or other associations of an organization to a product (see 9.1);
- product_related_product_category to classify a product with respect to specific criteria. One product_related_product_category is required to represent product type (see 1.1.3).

1.1.1.2 product_definition_formation

The product_definition_formation entity represents the identification of a specific version of the base product identification. A particular product_definition_formation is always related to exactly one product.

Each instance of product is required to have an associated instance of product_definition_formation. A single product entity may have more than one associated product_definition_formation. The set of these product_definition_formation entities represents the revision history of the product.

Attributes

- The ***id*** attribute uniquely identifies this version of the related product.
- The ***description*** attribute contains the reason for the creation of the version.

- The attribute ***of_product*** provides a link to the base product of which this entity represents a version.

ENTITY product_definition_formation	Attribute Population	Remarks
id	type: identifier = string part version identification	Unique in relation to the specified of_product
description	type: text = string	OPTIONAL
of_product	type: entity = product	

Preprocessor Recommendations: If an organization does not version parts, it is recommended that the id attribute contain the string '/NULL' to indicate that no version information is relevant or intended. In this case only a single product_definition_formation for the part is possible. Some preprocessors have used an id value of 'ANY' (or '/ANY') to indicate that *any* existing revision of a component is valid for the parent assembly. This technique may reduce the amount of data sent in change packages, but it also reduces the ability to track the actual contents of parts lists at a particular change level.

Postprocessor Recommendations: If the value of the id attribute for a product_definition_formation is the string '/NULL', postprocessors should use this as an indication that the sending system or business process does not support versioning of parts. Postprocessors may also recognize an id value of 'ANY' (or '/ANY') as a generic revision of a part when it is involved as a component in an assembly. This has been used to indicate that *any* existing revision of the component is valid for use the parent assembly.

Related Entities: In general, each product_definition_formation is recommended to have an associated product_definition representing the view definition for the part version. In restricted cases, a part version without a definition may be used to enhance information about another related, fully-defined version. In two specific cases a part version may be exchanged without an associated view definition:

1. When version history (sequence relationship) is represented - only the most recent version is required to have an assigned product_definition. If there is no product_definition associated with the previous versions, only basic information about the sequence of previous versions is exchanged as additional information about the current part version that is the focus of the data exchange;
2. Where a supplied item is identified using the alias relationship - the supplier part version may not have an associated product_definition.

In addition, a product_definition_formation may be optionally referenced by the following entities:

- product_definition_formation_relationship to associate two product versions with each other to characterize the version history;
- applied_organization_assignment (or applied_person_and_organization_assignment) to represent the update/modifier, customer, or other associations of an organization to a product version;
- applied_date_assignment (or applied_date_and_time_assignment) to represent various dates related to a product version;
- applied_approval_assignment to record the authorization status of a product version (see 9.2);
- applied_security_classification_assignment to record a level of security clearance for a product version.

1.1.1.3 product_definition_formation_with_specified_source

The product_definition_formation_with_specified_source entity is a subtype of the entity product_ definition_formation. This entity adds the attribute make_or_buy to indicate the source of the version - either 'made' or 'bought'. This entity is never used in the context of 'Document as Product', but it may be used in 'Part as Product' for compatibility with AP203. However, this make-versus-buy distinction can be ambiguous in the context of exchange - going down a supply chain, the sender is often 'bought' while the receiver is 'made', while in the other direction, the sender is 'made' and the receiver 'bought'. Because of this ambiguity the use of this entity is not generally recommended.

Attributes

- The ***make_or_buy*** attribute contains the source information.

ENTITY product_definition_formation_with_specified_source	Attribute Population	Remarks
id	Same as supertype.	Same as supertype.
description	Same as supertype.	Same as supertype.
of_product	Same as supertype.	Same as supertype.
make_or_buy	type : source (enumeration)	not recommended

Preprocessor Recommendations: The use of this subtype is optional. It is not recommended in the general case. The source value is an enumerated type having possible values '.MADE.', '.BOUGHT.', or '.NOT KNOWN.'. '.MADE.' indicates the part is built within the company. '.BOUGHT.' indicates a vendor part, but this distinction can be unclear when the data is exchanged.

Postprocessor Recommendations: If this entity is encountered, the postprocessor should at a minimum process the information from the supertype entity product_definition_formation.

Related Entities: There are no specific related entities.

1.1.1.4 product_definition

The product_definition entity represents the identification of a particular view on a version of the product base identification relevant for the requirements of particular life-cycle stages and application domains. View may be based on application domain and/or life-cycle stage (e.g., design, manufacturing). A view collects product data for a specific task. It is possible to have many product_definition views for a part/version combination.

The product_definition entity enables the establishment of many important relationships. Product_definition is the central element to link product information to parts, e.g., assembly structure, properties (including shape), and external descriptions of the product via documents.

The use of product_definition entities is not strictly required by rules in the PDM Schema, but it is strongly recommended. All product_definition_formation entities should always have at least one associated product_definition, except in the case of supplied product identification and version history information.

Attributes

- The ***id*** attribute identifies which view of the product the particular instance represents.
- The ***description*** attribute specifies the word or group of words used to refer to the product_definition.
- The ***formation*** attribute links to the product_definition_formation of which this represents a view.
- Context information is stored in the ***frame_of_reference*** attribute. All STEP product_definitions must be founded in some product_definition_context that identifies the application domain and life-cycle stage from which the data is viewed. (see 1.1.2.).

ENTITY product_definition	Attribute Population	Remarks
id	type: identifier = string	must be unique in relation with a specific product_version
description	type: text = string	OPTIONAL
formation	type: entity = product_definition_formation	reference to the associated product_definition_formation
frame_of_reference	type: entity = product_definition_context	reference to the associated product_definition_context (see 1.1.2.4)

Preprocessor Recommendations: There is no standard mapping for the id attribute of product_definition, however the value should be unique relative to others related to the same product_definition_formation. Previous use of the id attribute on product_definition had sometimes 'overloaded' the unique identifier with informal life-cycle or organization information. This is not generally recommended for the PDM Schema - this attribute should contain a unique identifier for the part view definition - no additional semantics are associated with this attribute in the PDM Schema..

Postprocessor Recommendations: Previous use of the id attribute on product_definition had sometimes 'overloaded' the unique identifier with informal life-cycle or organization information, this should not be expected in general or required for postprocessing.

Related Entities: A product_definition may be referenced by the following entities:

- applied_identification_assignment to assign alias identifiers,
- product_definition_relationship to associate two product_definitions with each other to characterize various product structures,
- property_definition to associate general properties,
- configuration_design to associate configuration identification information,
- applied_organization_assignment (or applied_person_and_organization_assignment) (see 9.1),
- applied_date_assignment (or applied_date_and_time_assignment) to represent various dates related to a product version,
- applied_approval_assignment to record the authorization status of a product version (see 9.2),
- applied_security_classification_assignment to record a level of security clearance for a product version,
- applied_document_reference to assign documents (external descriptions) to the product_definition.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#30 = PRODUCT_CONTEXT('', #20, '');
#40 = PRODUCT('part_id', 'part_name', 'part_description', (#30));
#60 =
PRODUCT_DEFINITION_FORMATION('pversion_id','pversion_description',
#40);
#80 = PRODUCT_DEFINITION('view_id', 'view_name', #60, #90);
#90 = PRODUCT_DEFINITION_CONTEXT('part definition', #20, '');
```

Example 1: exchange file segment for part master identification

1.1.2 Context Information

Context information provides a scope and necessary circumstance for product identification information. It consists of two separate and related areas:

- Application Protocol Identification,
- Application Context Information.

Application Protocol identification is provided by the entity application_protocol_definition. It characterizes the STEP Application Protocol or similar working draft STEP specification that includes or uses the PDM Schema and provides the scope for the exchange data set.

Application context information identifies the usage of the information within the scope of the PDM Schema. Application context information is divided into application domain information and product definition context information:

- The application domain is identified by the application_context entity. It provides a basis for the interpretation of the information represented in the product data exchange;
- The product_definition_context carries information distinguishing the life-cycle stage (e.g., design, manufacturing) relevant to a particular product view definition as well as indication of the type of the definition - e.g., physical, zonal, functional.

Identification of a valid scope and context of interpretation for the identifier of a product is also required in the PDM Schema. This involves the assignment of an organization to the product, in the role 'id owner' to identify the scope of uniqueness and validity of the product id.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of context information related to basic product identification are illustrated in Diagram 2 below. The corresponding STEP exchange file encoding is illustrated in Example 2.

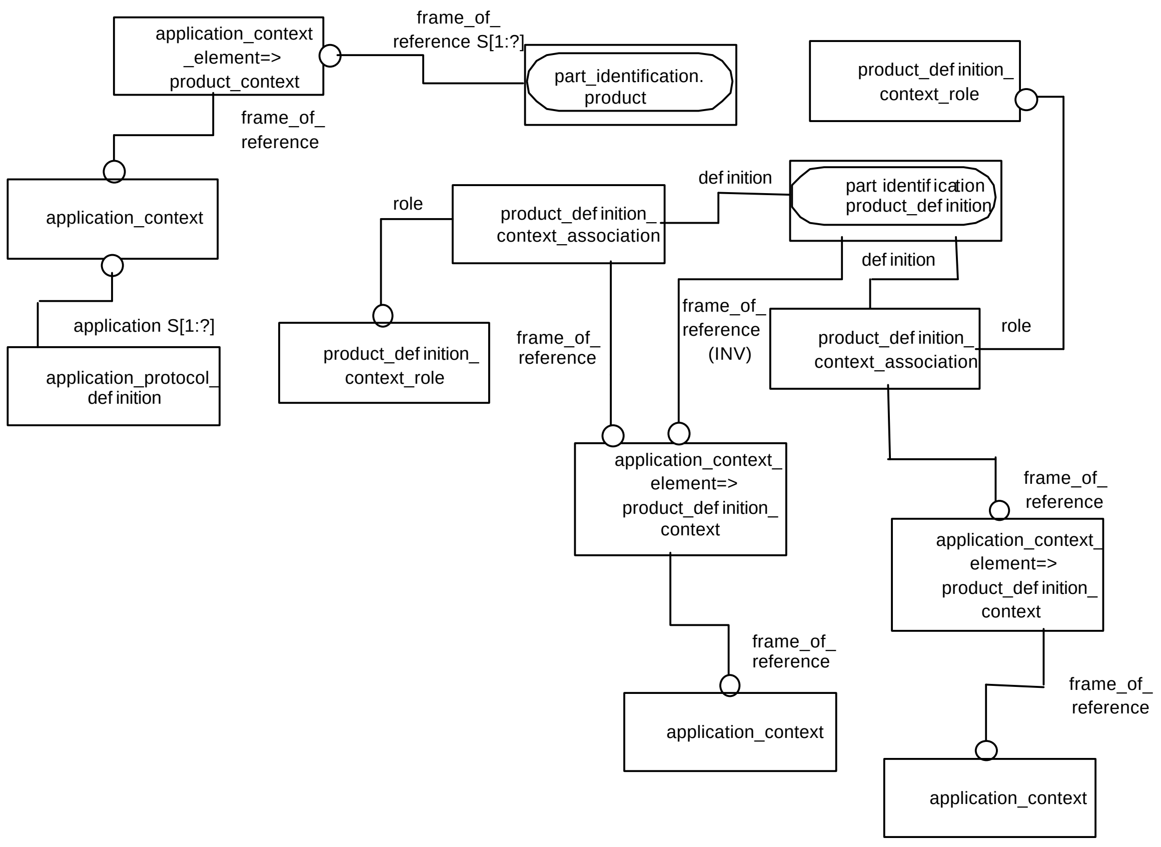

application_context _element=> product_context
frame_of_ reference S[1:?]
part_identification. product
product_definition_ context_role
frame_of_ reference
application_context
definition
part identification product_definition
role
product_definition_ context_association
definition
application S[1:?]
application_protocol_ definition
product_definition_ context_role
frame_of_ reference
frame_of_ reference (INV)
product_definition_ context_association
role
application_context_ element=> product_definition_ context
frame_of_ reference
application_context_ element=> product_definition_ context
frame_of_ reference
frame_of_ reference
application_context
application_context

Diagram 2: Part Master with Context Information Instance Diagram

1.1.2.1 application_protocol_definition

The application_protocol_definition entity represents the identification of the STEP Application Protocol that specifies the scope and extent of the application domain appropriate for this information model.
An AP that uses the PDM Schema should provide values for the attributes of this entity.

Attributes

- The ***application_interpreted_model_schema_name*** attribute identifies the name of the EXPRESS schema that specifies the information model used for the data representation and exchange.
- The ***application_protocol_year*** attribute identifies the year of publication of the specification.
- The ***status*** attribute provides information about the degree of maturity of the defining specification.
- The ***applications*** attribute provides a reference to the associated application_context entity.

ENTITY application_protocol_definition	Attribute Population	Remarks
application_interpreted_model_schema_name	type: label = text 'pdm_schema'	If the PDM Schema is contained within an Application Protocol, the AP name should be used.
application_protocol_year	type: year_number = integer 1998	If the PDM Schema is contained within an Application Protocol, the AP year should be used.
status	type: label = text 'version 1.1'	If the PDM Schema is contained within an Application Protocol, the AP status should be used.
applications	type: entity = application_context	

Preprocessor Recommendations: This entity should be instantiated exactly once in the data set. In some cases, an application protocol may support multiple application_contexts. The entity application_-protocol_definition should reference the 'primary' defining application context for the product data that is being exchanged.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

1.1.2.2 application_context

The application_context entity identifies the application domain that defined the data. The application_context entity may have an identifier associated with it through the entity id_attribute and its attribute_value attribute. The application_context entity may have a description associated with it through the entity description_attribute via the attribute attribute_value. It is not recommended to instantiate these optional values.

There exists a 'primary' application context for each product_definition. This is the value of the attribute product_definition.frame_of_reference; it is the frame_of_reference for the 'primary' product_definintion_context (see 1.1.2.4). This 'primary' application context represents the defining application domain for each product_definition. Additional application domains may be associated with a product_definition through additional product_definition_contexts via the entity product_definition_context_association.

Attributes

- The ***application*** attribute identifies the name of the general application domain that defined the data.

ENTITY application_context	Attribute Population	Remarks
application	type: label = text	

Preprocessor Recommendations: This entity should be instantiated once in the data set for each relevant application domain. Where appropriate, the value 'assembly study', 'digital mock up', 'process planning', 'electrical design', or 'mechanical design' should be used. Note that 'electrical design' and 'mechanical design' are not understood to mean the design life cycle - e.g., within the application domain 'mechanical design', there may be concurrent 'manufacturing' and 'design' life-cycle view definitions.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

1.1.2.3 product_context

The product_context entity is a subtype of application_context_element. All products in STEP must be founded in a product_context to specify the point of view of the application. The product_context entity identifies the engineering discipline's point of view from which the data is being presented. This entity will establish the context perspective and source of requirements for product entities (see 1.1.1.1).

Attributes

- The ***discipline_type*** attribute contains a description of the discipline point of view for a product.

ENTITY product_context	Attribute Population	Remarks
name	type: label = text	
frame_of_reference	type: entity = application_context	
discipline_type	type: label = text	

Preprocessor Recommendations: It is recommended to instantiate this entity exactly once in the data set. There is no standard mapping for the name attribute of a product_context. There is no standard mapping for the discipline_type attribute.

Postprocessor Recommendations: Past use of the attribute discipline_type may result in values such as 'mechanical'or'electrical'.

Related Entities: There are no specific related entities.

1.1.2.4 product_definition_context

The product_definition_context entity is a subtype of application_context_element. All product_definitions in STEP must be founded in a product_definition_context to identify the product definition type and life-cycle stage from which the data is viewed.

There are two ways to implement the product_definition_context in STEP:

- association of a single primary context with a product_definition,
- assignment of more than one (additional) context to a product_definition.

The value of the attribute product_definition.frame_of_reference is a product_definition_context that represents the 'primary' defining context for a view. In general this defining context is qualified both by type (product_definition_context.name) and by life-cycle (product_definition_context.life_cycle_stage) information. The primary context also has associated application domain information (see 1.1.2.2).

The product_definition_context_association entity allows for multiple additional contexts to be associated with a single product_definition. There is always one required 'primary' context that identifies the defining application domain and life-cycle information. Additional product_definition_context entities identify additional concurrent relevant views on the product_definition. These application_contexts are related to the product definition by product_definition_context_association.

Attributes

- The ***life_cycle_stage*** attribute identifies the life-cycle view on the associated product_definition.

ENTITY product_definition_context	Attribute Population	Remarks
name	type: label = text	Distinguishes the type of the associated product_definition
frame_of_reference	type: entity = application_context	
life_cycle_stage	type: label = text	string appropriate to describe the life-cycle point of view

Preprocessor Recommendations: The name attribute provides a distinction on the type of view on a part version ('part definition') from one of a document version ('digital document definition', 'physical document definition'). This attribute may also indicate other types of definitions: e.g., functional, or spatial and/or zonal.

The attribute life_cycle_stage contains a description of the particular viewpoint from which the data may be interpreted. Recommended values include 'design' and 'manufacturing'. Preprocessors should identify life-cycle information for the defining 'primary' product_definition_context, they may also indicate additional relevant life_cycle_stage information with a second product_definition_context, related via the product_definition_context_association. For AP214 compatibility, it is recommended that preprocessors generate an instance of product_definition_context_association relating a product_definition with the 'primary' product_definition_context.

Postprocessor Recommendations: Postprocessors should interpret the value of the name attribute as a type distinction between various definitions of parts and documents. The life_cycle_stage attribute value may be interpreted as the relevant viewpoint from which the data is valid. Interoperable processors will understand life-cycle information from the defining 'primary' product_definition_context as well as any additional life-cycle information provided by other relevant product_definition_context entities, related via the product_definition_context_association. The product_definition_context_association entity may or may not be present relating a product_definition with the 'primary' context information.

Related Entities: There are no specific related entities.

1.1.2.5 product_definition_context_association

The product_definition_context_association entity allows for multiple additional contexts to be associated with a single product_definition.

This entity is never used in the 'Document as Product' approach. With 'Part as Product', there is always one required 'primary' context that identifies the defining application domain and life-cycle information from which the data is viewed. This 'primary' context is the value of the attribute product_definition.frame_of_reference. For AP214 compatibility, it is recommended that an instance of product_definition_context_association exist relating a product_definition with its 'primary' context information.

Additional product_definition_context entities identify additional concurrent relevant views on the product_definition. These additional contexts are related to the product definition by the entity product_definition_context_association.

Attributes

- The ***definition*** attribute provides a reference to the associated product_definition entity.
- The ***frame_of_reference*** attribute is a pointer to the associated product_definition_context entity.
- The ***role*** attribute gives an optional role indication to the association.

ENTITY product_definition_context_ association	Attribute Population	Remarks
definition	type: entity = product_definition	
frame_of_reference	type: entity = product_definition_context	
role	type: entity = product_definition_context_role	

Preprocessor Recommendations: For AP214 compatibility, it is recommended that preprocessors generate an instance of product_definition_context_association relating a product_definition with the 'primary' product_definition_context.

Postprocessor Recommendations: Interoperable postprocessors should expect that a product_definition_context_association entity might be, or might not be, present to relate a product_definition with it's 'primary' context information.

Related Entities: There are no specific related entities.

1.1.2.6 product_definition_context_role

The product_definition_context_role entity provides a role string that is related to a product_ definition_context_association entity.

Attributes

- The ***name*** attribute provides the word or group of words by which the role is referred.
- The ***description*** attribute provides additional descriptive information related to the role.

ENTITY product_definition_context_role	Attribute Population	Remarks
name	type: label = string	
Description	type: text = string	OPTIONAL

Preprocessor Recommendations: There are no specific preprocessor recommendations.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#10 = APPLICATION_PROTOCOL_DEFINITION('version 1.1','pdm_schema',1998,
#20);
#20 = APPLICATION_CONTEXT('');
#30 = PRODUCT_CONTEXT('', #20, '');
#40 = PRODUCT('part_id', 'part_name', 'part_description', (#30));
#60 =
PRODUCT_DEFINITION_FORMATION('pversion_id','pversion_description',#40);
#80 = PRODUCT_DEFINITION('view_id', 'view_name', #60, #90);
#90 = PRODUCT_DEFINITION_CONTEXT('part definition', #20, '');
#100 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION (#80, #90, #110);
#110 = PRODUCT_DEFINITION_CONTEXT_ROLE ('', $);
```

Example 2 : exchange file segment for part master with context information

1.1.3 Type Classification

Type Classification information provides the basic capability to distinguish between products interpreted as *parts* and those interpreted as *documents*. This capability also supports distinguishing between different types of parts, e.g., *detail*, *assembly*, or *standard* parts.

A 'detail' is a 'part' **-** it represents a part that is not broken down into other identified, managed component parts in a product structure. The value 'detail' is used to distinguish from 'assembly' if this is required when exchanging product structure.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of part type classification are illustrated in Diagram 3.

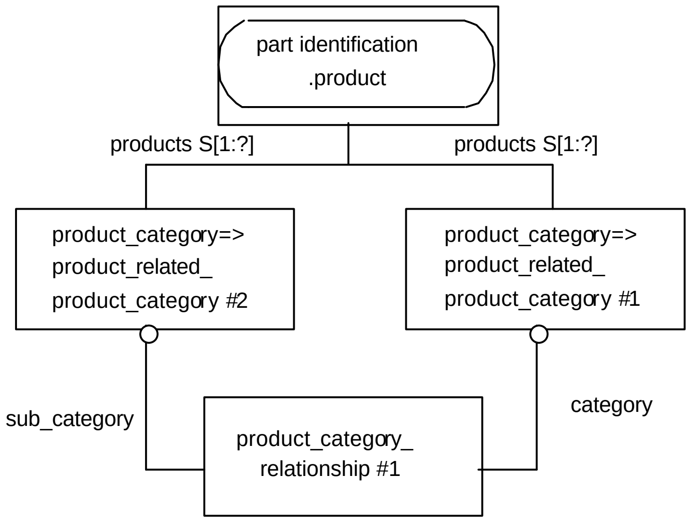

part identification
.product
products S[1:?]
products S[1:?]
product_category=> product_related_ product_category #2
product_category=> product_related_ product_category #1
sub_category
category
product_category_ relationship #1

Diagram 3: Part Master with Part Type Instance Diagram

1.1.3.1 product_related_product_category

The product_related_product_category entity is a subtype of product_category. It represents the identification of a specific classification applied directly to a product. The name and description attributes are inherited from the supertype product_category. This subtype adds the attribute products that allow it to be associated (related) directly to a product instance.

Attributes

- The ***products*** attribute associates the category with the product entity instances to which it applies.

ENTITY product_related_ product_category	Attribute Population	Remarks
name	type: label = text ‘part’, ‘detail’, ‘assembly’,	The primary value to use is 'part' **-** the values 'detail', 'assembly', and

	'standard'	'standard' indicate a further type distinction on the part.
description	type: text = string	OPTIONAL
products	type: entity = product	SET[1:?]

Preprocessor Recommendations: Preprocessors are recommended to always assign the type 'part' as a common distinction between parts and documents. Further possible type classifications like "tool" or "raw material" will be described in future releases in separate sections. Other values such as 'detail', 'assembly', or 'standard' may also be assigned, but should be related as sub-category to the primary 'part' using the product_category_relationship.

Postprocessor Recommendations: Postprocessors should recognize 'part' but also values of 'detail', 'assembly', and 'standard' as indicating the categorized product is a part.

Related Entities: There are no specific related entities.

1.1.3.2 product_category_relationship

The product_category_relationship entity establishes a category-subcategory relationship between two product_category entities.

Attributes

- The ***name*** attribute gives the word or group of words by which the relationship is referred.
- The ***description*** attribute provides additional descriptive information related about the relationship.
- The ***category*** attribute identifies the *super* category (more general) in the relationship.
- The ***sub_category*** attribute identifies the *sub* category (more specific) in the relationship.

ENTITY product_category_relationship	Attribute Population	Remarks
name	type: label = string	
description	type: text = string	
category	type: entity = product_related_product_category	
sub_category	type: entity = product_related_product_category	

Preprocessor Recommendations: There is no standard mapping defined for the name attribute. The description attribute is optional, and may contain any appropriate or mutually agreed upon string.
.
Postprocessor Recommendations: Postprocessors should expect the category value to be a product_related_product_category with name 'part'. The value for the attribute sub-category will be a product_related_product_category with name 'detail', 'assembly', or 'standard' to further qualify the type of part.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#160 = PRODUCT('H24-1123.1', 'Fixture RX25B', '', (#20));
#170 = PRODUCT('DIN 932', 'Screw M3x15', '', (#20));
#250 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#10, #160, #170));
#260 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #250, #270);
#270 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#160));
```

Example 3: exchange file for part master with type classification

The following section combines the above discussed concepts and segments in a complete example.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the complete requirements of part master identification with context and part type classification are illustrated in Diagram 4. This figure also illustrates the recommended assignment of an organization to the product in the role 'id owner' to identify the scope (context of interpretation) for uniqueness and validity of the product id. The corresponding STEP exchange file encoding for the complete part master is illustrated in Example 4.

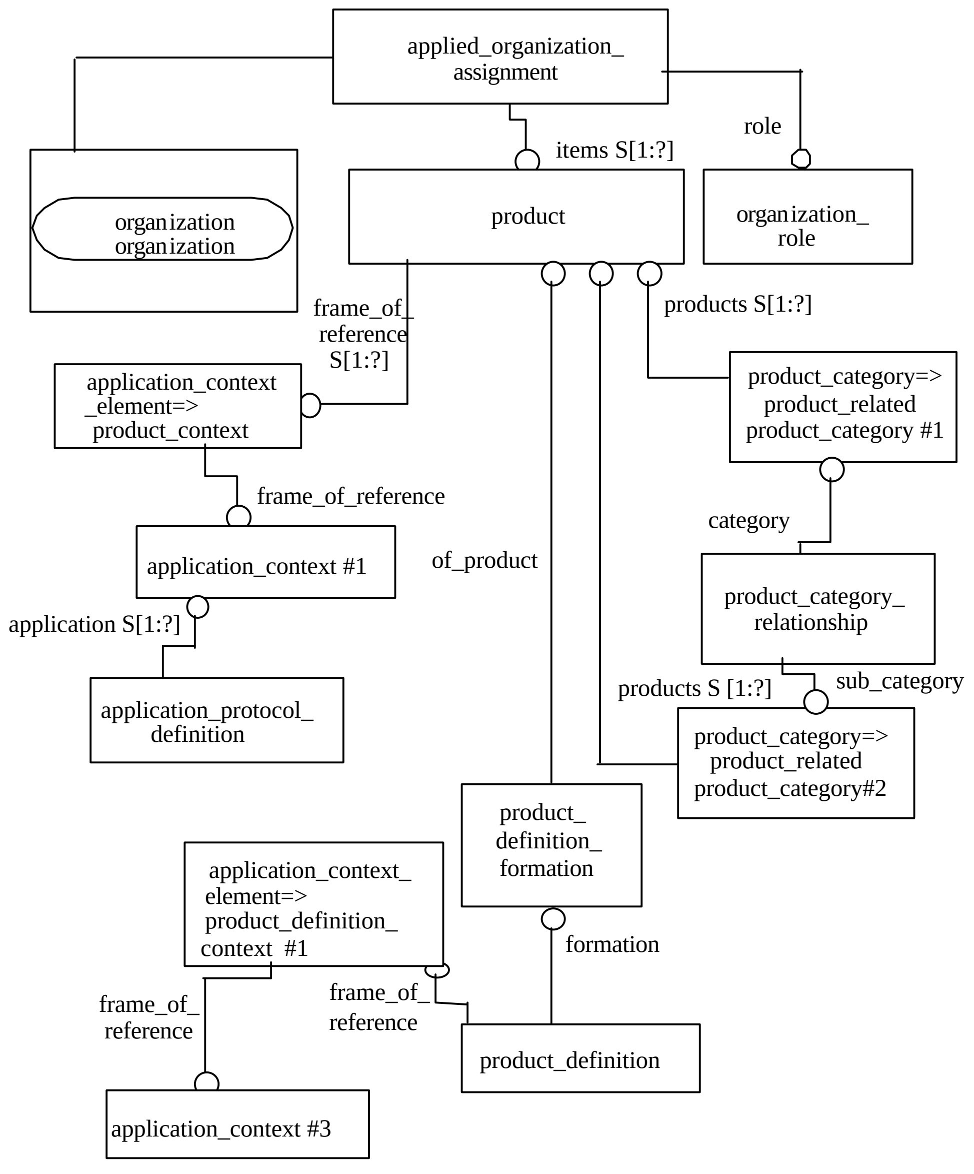


Diagram 4: Complete Part Master with Context and Type Classification Instance Diagram

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('test', 'file'), '2;1');
FILE_NAME('pid2_p21a.stp', '1999-05-03T21:03:29+00:00', ('N.N.'), (''),
'', '', '');
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
DATA;


/* part #1 */
#10 = PRODUCT('K01-42051', 'Bicycle Bell RX 3', $, (#20));

/* part context */
#20 = PRODUCT_CONTEXT('', #30, '');
#30 = APPLICATION_CONTEXT('');
#40 = APPLICATION_PROTOCOL_DEFINITION('version 1.1', 'pdm_schema',
1998, #30);

/* part versions for part #1 */
#50 = PRODUCT_DEFINITION_FORMATION('02', 'lever modified', #10);
#60 = PRODUCT_DEFINITION_FORMATION('03', 'upper housing modified',
#10);
#70 = PRODUCT_DEFINITION_FORMATION_RELATIONSHIP('', 'sequence', $, #50,
#60);

/* definition of view on version 03 of part #1 */
/* primary life_cycle_stage = design, primary application_domain =
mechanical design */
#80 = PRODUCT_DEFINITION('/NULL', $, #60, #90);
#90 = PRODUCT_DEFINITION_CONTEXT('part definition', #100, 'design');
#100 = APPLICATION_CONTEXT('mechanical design');

/* association of the id owner for part #1 */
#130 = APPLIED_ORGANIZATION_ASSIGNMENT(#140, #150, (#10, #160, #170));
#150 = ORGANIZATION_ROLE('id owner');

/* information on person and organization */
#140 = ORGANIZATION('ABC27166', 'Onyx AG', 'location');
#540 = PERSON('sarah.rijker@onyx.com', 'Rijker', 'Sarah', $, $, $);
#550 = PERSON_AND_ORGANIZATION(#540, #140);

/* part #2 and part #3 */
#160 = PRODUCT('H24-1123.1', 'Fixture RX25B', '', (#20));
#170 = PRODUCT('DIN 932', 'Screw M3x15', '', (#20));

/* part versions for part #2 and part #3 */
#180 = PRODUCT_DEFINITION_FORMATION('B', 'larger screw holes', #160);
#190 = PRODUCT_DEFINITION_FORMATION('15', '', #170);

/* view definition for version of part #2 */
#200 = PRODUCT_DEFINITION('/NULL', $, #180, #210);
#210 = PRODUCT_DEFINITION_CONTEXT('part definition', #215, 'design');
#215 = APPLICATION_CONTEXT('mechanical design');

/* view definition for version of part #3 */
#220 = PRODUCT_DEFINITION('/NULL', $, #190, #230);
```

```
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #240, 'design');
#240 = APPLICATION_CONTEXT('mechanical design');

/* part discriminator for parts #1 - #3 */
#250 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#10, #160, #170));
#260 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #250, #270);
#270 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#160));
#280 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #250, #290);
#290 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $, (#10));
#300 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #250, #310);
#310 = PRODUCT_RELATED_PRODUCT_CATEGORY('standard', $, (#170));

ENDSEC;
END-ISO-10303-21;
```

Example 4: exchange file for complete part master with context and type classification

1.2 Known Issues

1.2.1 application context information for product definition

The recommended mapping of the primary context perspective onto the product_definition_context referenced by product_definition.frame_of_reference conflicts with the current mapping of AP214. AP214 uses that context exclusively to assign product definition type information and requires the context defining the point of view on the product data to be captured in an additional product_definition_context that is attached via a product_definition_context_association. The usage guide chose an approach differing from AP214 DIS to ensure interoperability within the considered APs. On the other hand, this approach was chosen to allow a distinction between primary context and other contexts - a requirement that is currently not supported by 214 DIS. A ballot comment against AP214 DIS will be submitted.

2 Part properties

The PDM Schema allows specifying properties associated with parts. A property is the definition of a special quality and may reflect physics or arbitrary user defined measurements. A general pattern for instantiating property information is used in the PDM Schema. A number of pre-defined property type names are also proposed for use when appropriate.

A special case of part properties is that of the part shape property - a representation of the geometrical shape model of the part. Various relationships are defined between external geometrical models to represent geometric model structure and the associated transformation information required for digital mock-up of assembly structures.

2.1 General Part Properties

The PDM Schema allows specifying properties associated with product data by linking a representation of the property values to the object with which the property is associated.

The PDM Schema optionally allows property identification that is independent from the actual association of a property to product data. In this case, the property definitions of a given type are additionally associated to a single 'general' identification for the particular type of property. The general_property entity collects multiple property_definitions that are of the same type, which may be associated to different elements of product data.

2.1.1 Properties Associated with Product Data

The PDM Schema allows specifying properties associated with product data. To specify properties associated with product data, a property definition tree is instantiated that links a representation of the property values to the described object, that is the object with which the property is associated. The type of the property is given by the value of the attribute property_definition.name.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of property definitions associated with product data are illustrated in Diagram 5.

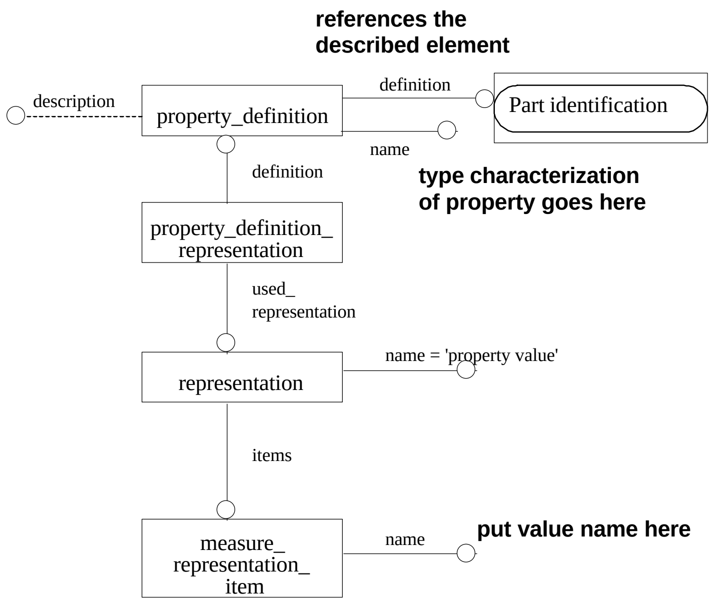


Diagram 5 : Property Definition Associated with Product Data Instance Diagram

2.1.1.1 property_definition

A property_definition is in the given context a property that characterizes a part. In the given context it applies to a product_definition that represents a life cycle and application domain specific view on a part.

ENTITY property_definition	Attribute Population	Remarks
name	type: identifier = string	Provides the property type name
description	type: text = string	OPTIONAL
definition	type: entity = here: product_definition	References product_definition as the element of part identification that is described by the property.

Pre-processor Recommendations: The name attribute characterizes the kind/type of the property. For the property_definition.name attribute, a number of predefined names for specific properties exist. Pre-processors should use these values where applicable.

Post-processor Recommendations: none specific

Related Entities: none specific

2.1.1.2 property_definition_representation

A property_definition_representation is in the given context an association between a property and its representation.

ENTITY property_definition_representation	Attribute Population	Remarks
definition	type: entity = property_definition	References associated property_definition
used_representation	type: entity = representation	References associated representation

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

2.1.1.3 representation

A representation in the context of document properties is a collection of one or more descriptive_representations_items that are related in a representation_context with type 'document parameters'. Herein a representation represents a document property.

ENTITY representation	Attribute Population	Remarks
name	type: label = STRING	shall be instantiated with 'property value'
items	type: entity = descriptive_representation_-item	the items that constitute the representation of the document property
context_of_items	type: entity = representation_context	is required to point to a representation_context with representation_context.type set to 'document parameters'

Pre-processor Recommendations: The name attribute shall be instantiated as 'property value' to indicate that the representation relates to a property value.

Post-processor Recommendations:. none specific

Related Entities: none specific

2.1.1.4 representation_context

The representation_context defines the context of interpretation for the values of items in a representation.

Attributes

- The ***context_identifier*** attribute identifies the context.
- The ***context_type*** attribute specifies the type of the context.

ENTITY geometric_representation_context	Attribute Population	Remarks
context_identifier	type: label = string	
context_type	type: text = string	should be instantiated with value 'document parameters'

Pre-processor Recommendations: In order to distinguish the use of a representation for document properties, the context_type attribute of the representation_context entity has the value 'document parameters'.

Post-processor Recommendations: none specific

Related Entities: none specific

2.1.1.5 measure_representation_item

A descriptive_representation_item is in the document property context a textual element that participates in one or more representations to define the respective properties.

ENTITY measure_representation_item	Attribute Population	Remarks
name	type: label = STRING	the name attribute indicates the name of the represented property
value_component	type: select = measure_value	
unit_component	type: entity = unit	

Pre-processor Recommendations: For the unit_component it is recommend to do complex instantiations of named_unit with si_unit or named_unit with conversion_based_unit

Post-processor Recommendations: none specific

Related Entities:. none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* Entities #5020 to #5070 assert that the in #290 defined view */
/* of part has a mass of 3KG                                    */
#5020 = PROPERTY_DEFINITION('', 'mass property', #290);
#5030 = PROPERTY_DEFINITION_REPRESENTATION(#5020, #5040);
#5040 = REPRESENTATION('property value', (#5060), #5050);
#5050 = REPRESENTATION_CONTEXT('', '');
#5060 = MEASURE_REPRESENTATION_ITEM('mass', MASS_MEASURE(3000), #5070);
#5070 =(NAMED_UNIT(*) MASS_UNIT() SI_UNIT($,.GRAM.));
```

Example 5 : exchange file segment for property definition associated with product data

2.1.2 Independent Property Identification

The PDM Schema also optionally allows property identification that is independent from the actual association of the property with product data. In this case, the property definitions of a given type are additionally associated with a single 'general' identification for the particular type of property.

The general_property entity collects multiple property_definitions of the same type, by usage of the general_property_association entity. Multiple property definitions of the same type may exist and be associated with different elements of product data. The relationship between a general_property identification and a related property also indicates if the related property is 'definitional' (i.e. can be used to discriminate a given element against others) or 'non-definitional'.

The PDM Schema also supports the specification of relationships between two general property objects. These relationships may be used to indicate that the value of one property can be derived from the value of another property. The relationship is modeled using the entity general_property_relationship.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of independent property identification is illustrated in Diagram 6.

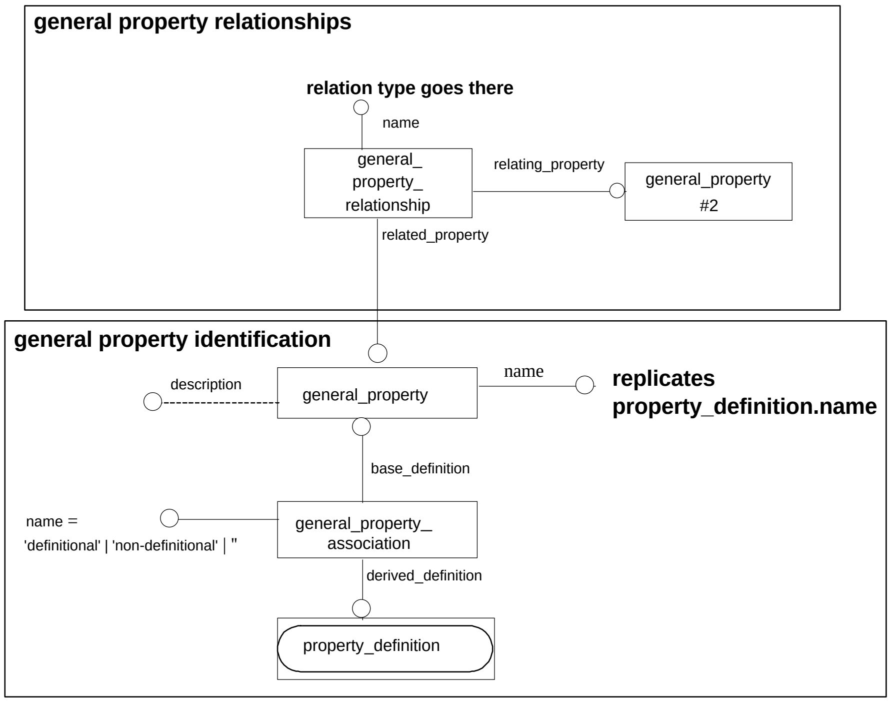

general property relationships
relation type goes there
name
general_ property_ relationship
relating_property
general_property #2
related_property
general property identification
description
general_property
name
replicates property_definition.name
base_definition
name = 'definitional' | 'non-definitional' | ''
general_property_ association
derived_definition
property_definition

Diagram 6 : Independent General Property Identification and Relationship Insance Diagram

2.1.2.1 general_property

A general_property identifies a property type/classification independently of the association of a definition of that type of property to product data.

Attributes
- The ***id*** attribute provides an identification of a general property.
- The ***name*** attribute stores the type of the described property.
- The ***description*** attribute can be used to provide a further description of the property type.
-

ENTITY general_property	Attribute Population	Remarks
id	type: identifier = string	recommended to be instantiated as ''
name	type: label = string	characterizes the property
description	type: text = string	OPTIONAL, provides further description of the property

Pre-processor Recommendations: General_property.name shall be replicated from the name attribute from the associated property_definition(s). For general_property.name a number of predefined names for specific properties exist. Pre-processors should use these values where applicable. The instantiation of a gneral_property in association to a property_definition is not mandatory.

Post-processor Recommendations: none specific

Related Entities: General_property is to be associated via general_property_association to the characterized property_definition.

2.1.2.2 general_property_association

A general_property_association associates a given property with a general_property in order to collect property definitions of the same type. The entity also carries the information whether a given property can be used to distinguish the described object from others.

Attributes

- The ***name*** attribute indicates whether the described property can be used to distinguish the described element from others of the same kind.
- The ***description*** attribute can be used to provide a further description of the property type.
- The ***base_definition*** attribute references the general_property that is associated with the property_definition.
- The ***derived_definition*** references the characterized property_definition.

ENTITY general_property_association	Attribute Population	Remarks
name	type: label = string	Specifies whether the associated Property_value object may be used to distinguish the described element from others of the same kind. A value of 'definitional' indicates that the associated Property_value distinguishes it from others. The permissive list for the name attribute is 'definitional', 'non-definitional' or ''.
description	type: text = string	OPTIONAL
base_definition	type: entity = general_property	
derived_definition	type: entity = property_definition	

Pre-processor Recommendations: the value of the name attribute may be used to indicate if the associated derived_definition is a defining and distinguishing property for the part definition that is described by the property.

Post-processor Recommendations: none specific

Related Entities: General_property_association relates a property_definition and a general_property.

2.1.2.3 general_property_relationship

A general_property_relationship asserts a relationship between two independently identified general property identifications. A typical example of the use of this entity is to describe the 'derived' relation type, where the relating_property is specified as being derived from the related_property.

Attributes

- The ***name*** attribute is used to describe the type of relationship between the properties.
- The ***description*** attribute can be used to provide optional further textual description.
- The ***relating_property*** attribute references the first of the related property pair.

- The ***related_property*** attribute references the second of the related property pair.

ENTITY general_property_relationship	Attribute Population	Remarks
name	type: label = string	
description	type: text = string	OPTIONAL
relating_property	type: entity = general_property	
related_property	type: entity = general_property	

Pre-processor Recommendations:. The meaning of the relating_property and related_property attributes is specified further by the relationtype indicated with general_property_relationship.name

Post-processor Recommendations: none specific

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* Entities #5000 to #5070 assert that the in #290 defined view */
/* of part2 has a mass of 3KG                                   */
#5000 = GENERAL_PROPERTY('', 'mass property', '');
#5010 = GENERAL_PROPERTY_ASSOCIATION('non-definitional', '', #5000,
#5020);
#5020 = PROPERTY_DEFINITION('', '', #290);
#5030 = PROPERTY_DEFINITION_REPRESENTATION(#5020, #5040);
#5040 = REPRESENTATION('property value', (#5060), #5050);
#5050 = REPRESENTATION_CONTEXT('', '');
#5060 = MEASURE_REPRESENTATION_ITEM('mass', MASS_MEASURE(3000), #5070);
#5070 =(NAMED_UNIT(*) MASS_UNIT() SI_UNIT($,.GRAM.));
```

Example 6 : exchange file segment for independent property identification

2.1.3 Pre-defined properties

For the representation of part properties, a number of property types have been pre-defined. It is recommended to use these values for the property_definition.name attribute where applicable.

- **Recyclability property -** A recyclability property is information concerning the ability to reuse objects or components of objects after their primarily intended usage.
  For recyclability properties of parts property_definition.name **= 'recyclability property'** shall be used
- **Mass property -** a mass property is a quantity of matter that an object consists of.
  For mass properties of parts property_definition.name **= 'mass property'** shall be used
- **Quality property -** A quality property is a property that enables to provide information about the level of quality of products or processes. A quality property documents e.g. the level of quality reached for the parts prototypes. For quality properties of parts property_definition.name **= 'quality property'** shall be used
- **Cost property -** a cost property is a property that specifies costs. For cost properties of parts property_definition.name **= 'cost property'** shall be used
- **Duration property -** A duration property is a property that specifies a period of time during which a given object is used or will last. For duration properties property_definition.name **= 'duration property'** shall be used

2.2 Part shape

The PDM Schema generally represents part master data. This part 'meta' data does not typically represent the detailed geometry of the part shape **-** this is the scope of CAD systems. Rather, the externally defined geometry is identified as an external representation of the part shape that is related to the part master data.

Part geometry is identified in the PDM Schema as a representation of a property of a part definition. The external part shape represents an external model referring to an external file that is associated with the PDM master file.

2.2.1 Geometric Shape Property

Part geometry is identified in the PDM Schema as a representation of a property of a part definition. As with general part properties, a representation of a property is identified and linked to the product definition by the entity property_definition. For the part shape property, property_definition is specialised to the subtype product_definition_shape.

The external part shape is represented by the entity shape_representation, a subtype of representation used in general part properties. Shape_representations model external models that are referenced in files associated with the PDM master file.

STEP Application Protocols define various subtypes of shape_representations with differing constraints on the allowable representation_items to explicitly represent the detailed geometric model. In the context of the PDM Schema, it is generally assumed that these subtypes of shape_representation are not used.

In the scope of the PDM Schema it is generally assumed that shape_representations are placeholders for externally defined geometry. This geometry may be defined in STEP format as well as in native CAD format. In general it is not recommended to instantiate dedicated geometric elements as items of the shape_representations for PDM Schema files. However, certain detailed elements of the shape are required to be able to place and relate the external geometric models together. Therefore, placement information is placed in the set of items of each shape_representation. Placement information is modeled using the entity axis_placement a subtype of geometric_representation_item.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of independent property identification is illustrated in Diagram 7.

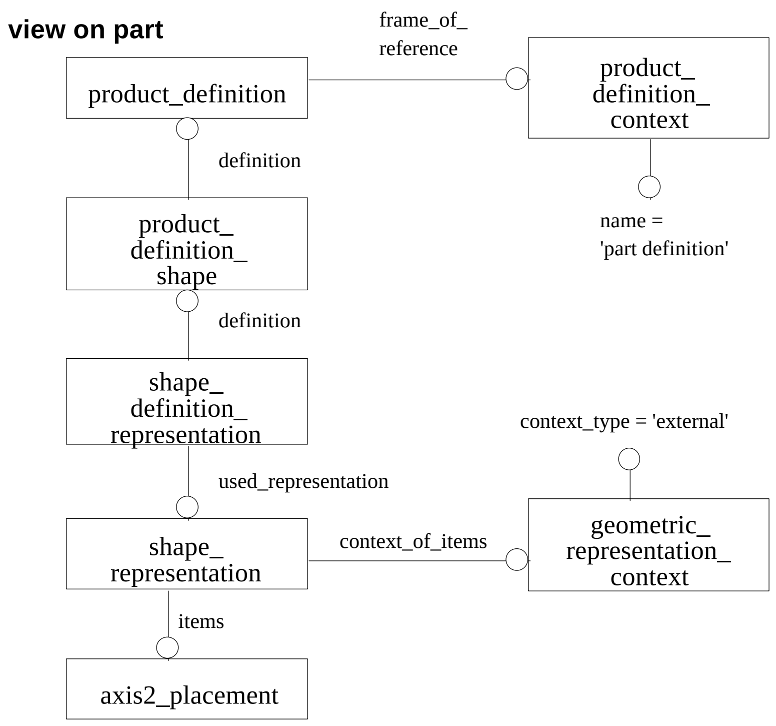


Diagram 7 : Assignment of Part Shape to the View of the Part Instance Diagram

2.2.1.1 product_definition_shape

Product_definition_shape represents the shape of a product. The shape may be a conceptual shape for which a specific geometric representation is not required.

ENTITY product_definition_shape	Attribute Population	Remarks
name	type: label = string	no standard mapping exists should be instantiated as empty string
description	type: text = string	no standard mapping exists should be instantiated as empty string
definition	type: entity = product_definition	reference to an instance of product_definition to which the shape information is attached

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

2.2.1.2 shape_definition_representation

A shape_definition_representation links the definition of a shape, i.e. the product_definition_shape, with its representation, i.e. a shape_representation.

For a given product_definition_shape alternative representations might be given by multiple occurrences of shape_definition_representation that each link a representation to one product_definition_shape.

Attributes

- The ***definition*** attribute is used to point to the product_definition_shape for which the representation of shape is provided.
- The ***used_representation*** references an instance of shape_representation that provides a representation of the part shape.

ENTITY shape_definition_representation	Attribute Population	Remarks
definition	type: entity = product_definition_shape	
used_representation	type: entity = shape_representation	

Pre-processor Recommendations: Some APs allow the assignment of a role ('detailed representation' or 'idealized representation') to a shape_definition_representation by using *name_attribute*.named_item. However, it is recommended to use a document content property (see 8.2) to record this type of information instead of using *name_attribute*.attribute_value

Post-processor Recommendations: none specific

Related Entities: none specific

2.2.1.3 shape_representation

Shape_representation is a subtype of representation that is specifically used to represent shape information.

Attributes

- The ***name*** attribute unambiguously identifies the geometric model within the CAD environment.
- The ***items*** collect the items of the shape_representation.
- The ***context_of_items*** attribute references a geometric_representation_context that establishes the coordinate space for the shape_representation.

ENTITY shape_representation	Attribute Population	Remarks
name	type: label = string	
items	type: set of representation items	a set of *representation_item*s that are related in the context_of_items
context_of_items	type :entity = geometric_representation_context	reference to an instance of *geometric_representation_context* or an complex instance of (*geometric_representation-_context, global_uncertainty-_assigned_context*)

Pre-processor Recommendations: none specific.

Post-processor Recommendations: none specific

Related Entities: none specific

2.2.1.4 geometric_representation_context

The geometric_representation_context defines the coordinate space in which the geometric model defined by a shape_representation resides.

Attributes
- The ***context_identifier*** attribute identifies the coordinate system.
- The ***context_type*** attribute specifies the type of context.
- The ***coordinate_space_dimension*** attribute defines the dimensionality of the coordinate space.

ENTITY geometric_representation_context	Attribute Population	Remarks
context_identifier	type: label = string	
context_type	type: text = string	should be instantiated with 'external'
coordinate_space_dimension	type: dimension_count = INTEGER	it is recommended to instantiate with 3

Pre-processor Recommendations:. The explicit representation of geometry is not in the scope of the PDM Schema. The PDM Schema supports exclusively shape_representation that are externally defined. Thus the context_type attribute shall be instantiated with 'external'.

Post-processor Recommendations: none specific

Related Entities: geometric_representation_context specifies the coordinate system for the use by shape_representations.

2.2.1.5 geometric_representation_item

Geometric_representation_items are items for product data representation with the additional meaning of having geometric context. Geometric_representation_item are subtyped to a variety of entities in the context of geometric representations. The subtype relevant for the PDM Schema are axis_placement that can be used to define transformations between shape_representations.

Attributes
- The ***name*** attribute is used to name the geometric_representation_item.

ENTITY geometric_representation_item	Attribute Population	Remarks
name	type: label = string	

Pre-processor Recommendations: only subtype of geometric_representation_item should be instantiated

Post-processor Recommendations: none specific

Related Entities: most relevant in the scope of the PDM Schema are axis2_placement_3d to define transformation information.

2.2.1.6 axis2_placement_3d

Axis2_placement_3d is a geometric_representation_item that specifies the location and orientation in three-dimensional space of two mutually perpendicular axes.

Attributes

- The ***name*** attribute provides a name for the placement.
- The ***location*** attribute defines the spatial position of the reference point and origin of the associated placement coordinate system.
- The ***axis*** attribute defines the exact direction of the local Z axis.
- The ***reference_direction*** attribute can be used to determine the direction of the local X axis.

ENTITY axis2_placement_3d	Attribute Population	Remarks
name	type: label = string	
location	type: entity = cartesian_point	
axis	type: entity = direction	OPTIONAL
reference_direction	type: entity = direction	OPTIONAL

Pre-processor Recommendations: the meaning of the relating_property and related_property attributes is specified further by the relation type indicated with general_property_relationship.name

Post-processor Recommendations: If necessary an adjustment to ref_direction has to be made to maintain orthogonality to the axis direction. If axis or ref_direction are omitted, these directions are taken from the geometric coordinate system.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#290 = PRODUCT_DEFINITION('p2v', 'view on part2', #280, #130);
/* Entities #600 to #660 define the geometric model for 'part2' */
#600 = PRODUCT_DEFINITION_SHAPE('shape_p2',$,#290);
#610 = SHAPE_DEFINITION_REPRESENTATION(#600,#620);
#620 = SHAPE_REPRESENTATION('sol2',(#670),#630);
#630 = GEOMETRIC_REPRESENTATION_CONTEXT('c2','external',3);

/* Entities #640 to #670 are the elements of shape representation */
#640 = CARTESIAN_POINT('', (3.0E+001, 0.0E+000, 1.0E+001));
#650 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#660 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#670 = AXIS2_PLACEMENT_3D('', #640, #650, #660);
```

Example 7 : exchange file segment to associate shape with part views

2.2.2 Portions of the Part Shape

The PDM Schema supports the requirement to identify explicitly a portion of shape that has to be associated with other information – including dedicated geometric models. Property information can be linked to shape aspects via property definitions. A prominent special case is the assignment of dedicated shape_representations as shown in the diagram below.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of independent property identification are illustrated in Diagram 8.

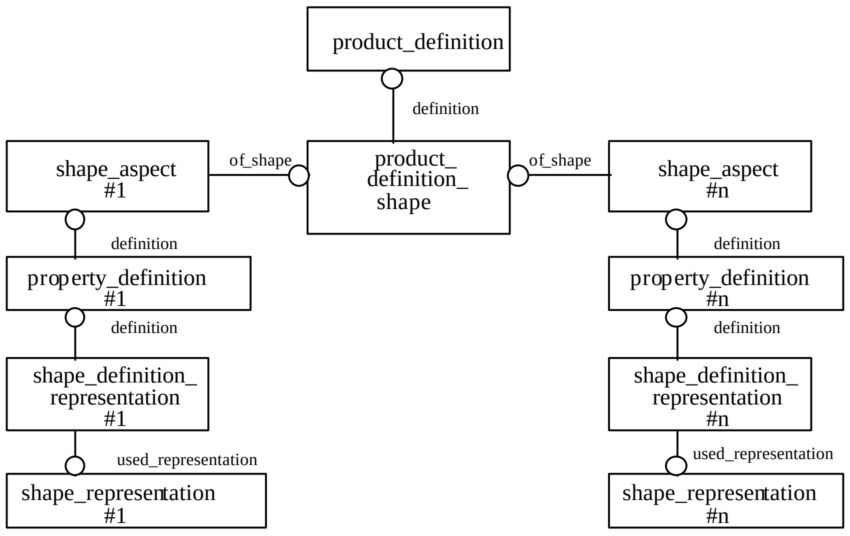


Diagram 8 : Identification and Representation of Portions of Part Shape Instance Diagram

2.2.2.1 shape_aspect

Attributes

- The ***name*** is the organizational name of the shape_aspect.
- The ***description*** attribute provides an optional textual description of the shape_aspect.
- The ***of_shape*** attribute references the product_definition_shape of which the shape_aspect is a portion.
- The ***product_definitional*** attribute indicates whether the portion of shape lies on the physical boundary of the part shape.

ENTITY shape_aspect	Attribute Population	Remarks
name	type: label = STRING	
description	type: text = STRING	OPTIONAL
of_shape	type: entity = product_definition_shape	
product_definitional	type: LOGICAL	If a value of TRUE is given, then it is asserted that the shape_aspect is on the physical boundary of the product_definition_shape.

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#700 = PRODUCT_DEFINITION_SHAPE('shape_p',$,#260);
#710 = SHAPE_ASPECT('aspect1',$,#700,.T.);
/* Entities #730 to #790 define the geometry related to  */
/* shape_aspect #710                                     */
#730 = PROPERTY_DEFINITION('shape for aspect1',$,#710);
#740 = SHAPE_DEFINITION_REPRESENTATION(#730,#750);
#750 = SHAPE_REPRESENTATION('sa1',(#795),#760);
```

Example 8: exchange file segment for identification of shape portions

2.2.3 Relating Externally Defined Part Shape to an External File

The PDM Schema generally represents part master data. This part 'meta' data does not typically represent the detailed geometry of the part shape - this is the scope of CAD systems. Rather, the externally defined geometry is identified as an external representation of the part shape that is related to the part master data. Part geometry is identified in the PDM Schema as a representation of a property of a part definition - it is actually in externally referenced files. These files are often generated by CAD systems, and contain the detailed geometric shape representation with dedicated geometric elements.

The external part shape is an external geometric model related to the part master data and is referenced as an external CAD file. Specifics of external file identification are described in Section 5. The external file that contains the detailed geometry is attached to the shape_representation representing the external geometric model using the entities property_definition and property_definition_representation.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of relating part shape to the external CAD file is illustrated in Diagram 9.

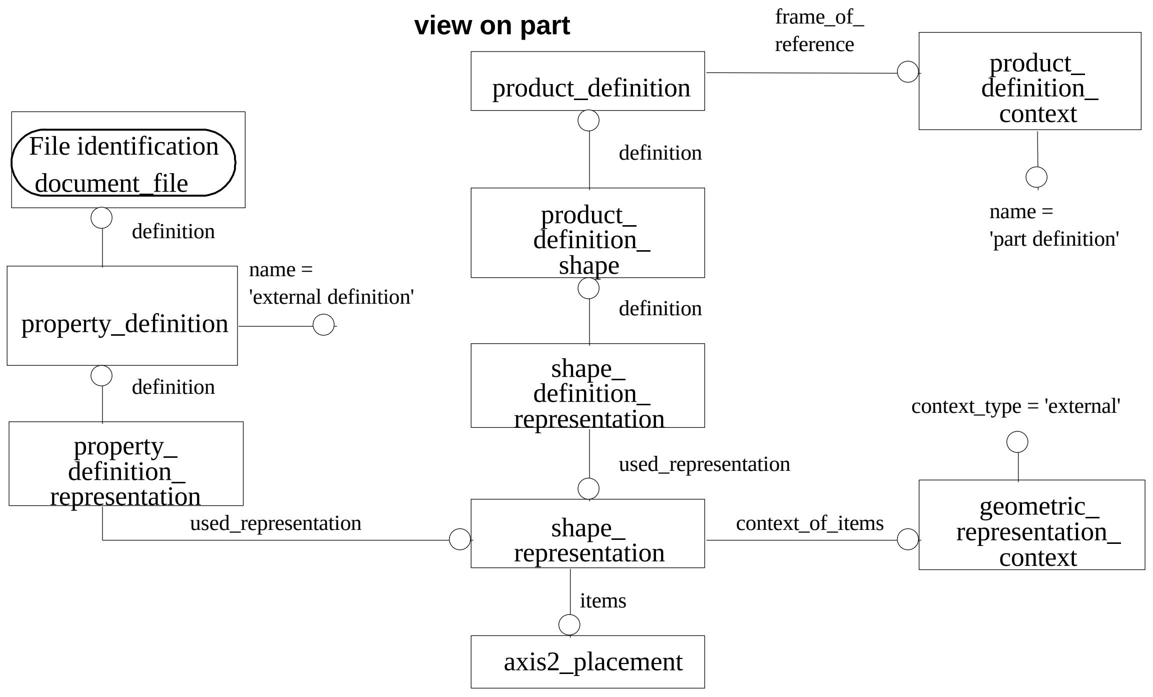


Diagram 9: Part Shape Related to External File Instance Diagram

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* Entities #370 to #497 define a managed document which */
/* contains a representation of the shape of 'part2'    */
#370 = PRODUCT('p2_shape_doc', 'document reflecting
         shape of part2', '', (#20));
#380 = PRODUCT_RELATED_PRODUCT_CATEGORY('document', '', (#370));
#390 = DOCUMENT_TYPE('configuration controlled document version');
#400 = DOCUMENT('', '', '', #390);
#410 = APPLIED_DOCUMENT_REFERENCE(#290, 'equivalence', (#400));
#420 = OBJECT_ROLE('mandatory', '');
#430 = ROLE_ASSOCIATION(#420, #410);
#440 = PRODUCT_DEFINITION_FORMATION('1', '', #370);
#450 = DOCUMENT_PRODUCT_EQUIVALENCE('equivalence', '', #400, #440);
#460 = PRODUCT_DEFINITION_CONTEXT('digital document definition', #10,
'';
#470 = PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS('p2_shape_d',
   'digital document for part2 shape', #440, #460,(#475));
#475 = DOCUMENT_FILE('p2_shape_file', '', '', #485, ' ', ' ');
#480 = DOCUMENT_REPRESENTATION_TYPE('digital',#475);
#485 = DOCUMENT_TYPE('');
#490 = IDENTIFICATION_ROLE('document_source',$);
#495 = APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT('',#490,#497,(#475));
#497 = EXTERNAL_SOURCE('part1_geometry.stp');
```

```
/* Entities #600 to #660 define the geometric model for 'part2' */
#600 = PRODUCT_DEFINITION_SHAPE('shape_p2',$,#290);
#610 = SHAPE_DEFINITION_REPRESENTATION(#600,#620);
#620 = SHAPE_REPRESENTATION('sol2',(#670),#630);
#630 = GEOMETRIC_REPRESENTATION_CONTEXT('c2','external',3);
#640 = CARTESIAN_POINT('', (3.0E+001, 0.0E+000, 1.0E+001));
#650 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#660 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#670 = AXIS2_PLACEMENT_3D('', #640, #650, #660);

/* Entities #680 and #690 indicate that the shape_representation #620 /
/* is externally defined by the file represented by #475          */
#680 = PROPERTY_DEFINITION('external definition',$,#475);
#690 = PROPERTY_DEFINITION_REPRESENTATION(#680,#620);
```

Example 9: exchange file segment for externally defined geometry in managed documents

```
/* Entities #500 to #560 define the geometric model for 'part1' */
#500 = PRODUCT_DEFINITION_SHAPE('shape_p1',$,#230);
#510 = SHAPE_DEFINITION_REPRESENTATION(#500,#520);
#520 = SHAPE_REPRESENTATION('sol1',(#570),#530);
#530 = GEOMETRIC_REPRESENTATION_CONTEXT('c1','external',3);
#540 = CARTESIAN_POINT('', (1.0E+001, 0.0E+000, 1.0E+001));
#550 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#560 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#570 = AXIS2_PLACEMENT_3D('', #540, #550, #560);

/* Entities #575 to #585 state that the shape_representation #520 */
/* is externally defined in a digital file with the location */
/* 'part2_geometry.stp'                                      */
#575 = APPLIED_DOCUMENT_REFERENCE(#230,(#576));
#576 = DOCUMENT_FILE('p1_shape','',$,#577,'','');
#577 = DOCUMENT_TYPE('geometry');
#578 = ROLE_ASSOCIATION(#579,#575);
#579 = OBJECT_ROLE('mandatory',$);
#580 = DOCUMENT_REPRESENTATION_TYPE('digital',#576);
#581 = IDENTIFICATION_ROLE('document_source',$);
#582 = APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT('',#581,#583,(#576));
#583 = EXTERNAL_SOURCE('part2_geometry.stp');
#584 = PROPERTY_DEFINITION('external definition',$,#576);
#585 = PROPERTY_DEFINITION_REPRESENTATION(#584,#520);
```

Example 10: exchange file segment for externally defined geometry in flat files

2.3 Relationship Between Elements of Part Shape

The relative orientation and location of two shape_representations to each other can be defined with the shape_representation_relationships that reference transformations. One might want to specify such transformations to geometrically relate representations of portions of the part shape to relate the shape of components in an assembly structure in order to define the geometric relations.

The PDM Schema supports the definition of relationships between geometric models either:

- implicitly by defining two items that establish a reference relationship via an item_defined_transformation, or
- explicitly with the help of a functionally_defined_transformation. It is generally recommended that cartesian_transformation_operator is used to define a functionally_defined_transformation.

2.3.1 Implicitly defined transformations between geometric models

A relationship can be defined implicitly by referencing an item_defined_transformation that has two reference points to specify origin and target of the transformation.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of implicit definition of transformation between the elements of two representations is illustrated in Diagram 10.

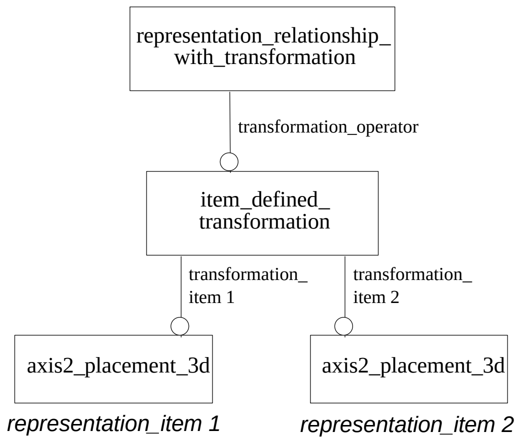


Diagram 10: Implicitly Defined Geometric Relations Instance Diagram

2.3.1.1 item_defined_transformation

An item_defined_transformation models a transformation performed by defining two representation_items before and after applying the transformation function. The transformation function is not explicitly provided, but it is derived through its relationship to the representation_items.

Attributes
- The ***name*** attribute provides a name for the transformation.
- The ***description*** attribute can be used to provide optional further textual description.
- The ***transform_item_1*** references the first item (origin for the transformation).
- The ***transform_item_2*** attribute references the second item (target for the transformation).

ENTITY item_defined_transformation	Attribute Population	Remarks
name	type: label = string	
description	type: text = string	OPTIONAL
transform_item_1	type: entity = representation_item	here: axis2_placement_3d
transform_item_2	type: entity = representation_item	here: axis2_placement_3d

Pre-processor Recommendations: To be meaningful the item_defined_transformation with axis2_placement_3d requires that the representations that include transfrom_item_1 and transform_item_2 share the same geometric_representation_context.

Post-processor Recommendations: none specific

Related Entities: none specific

2.3.2 Explicitly defined transformations between geometric models

Relationships between geometric models can be made in an explicit manner by using instances of cartesian_transformation_operator.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of explicit definition of transformation between the elements of two representations is illustrated in Diagram 11, Diagram 12, and Diagram 13.

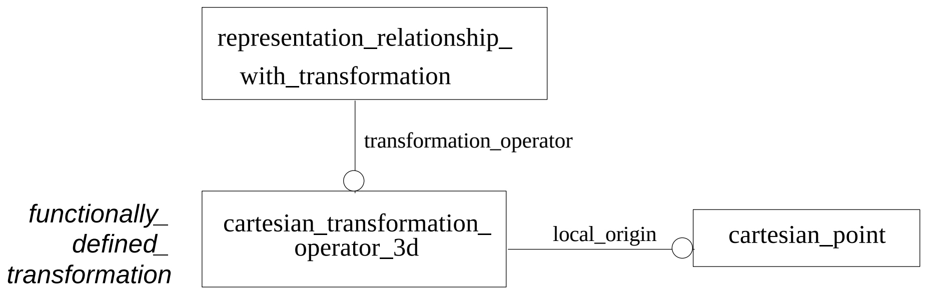

representation_relationship_ with_transformation
transformation_operator
functionally_ defined_ transformation
cartesian_transformation_ operator_3d
local_origin
cartesian_point

Diagram 11: Explicit Representation of a Translation Instance Diagram

The scale attribute with a positive value different from 1.0 allows uniform scaling in addition to translation or identity.

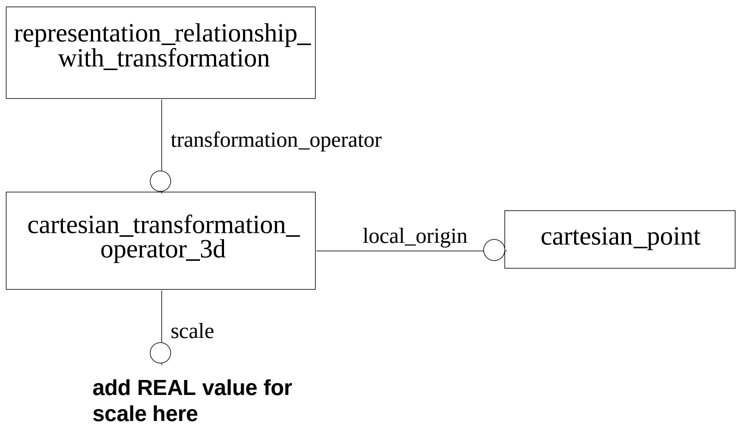

representation_relationship_ with_transformation
transformation_operator
cartesian_transformation_ operator_3d
local_origin
cartesian_point
scale
add REAL value for scale here

Diagram 12: Explicit Representation of a Translation with Scaling Instance Diagram

An additional rotation or mirroring requires the definition of the three **axis** attributes. If all optional attributes are set, a composition of all kinds of transformation is possible.

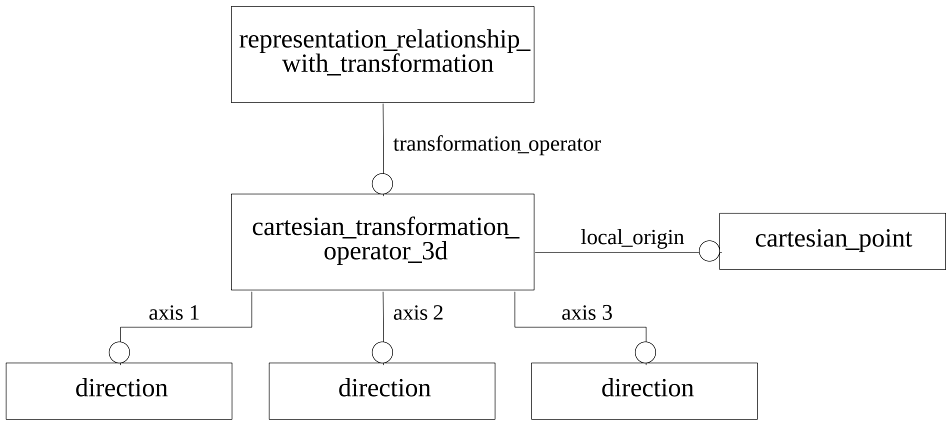


Diagram 13: Explicit Representation of a Translation with Rotation Instance Diagram

2.3.2.1 cartesian_transformation_operator

Cartesian_transformation_operator defines a geometric transformation composed of translation and rotation. Mirroring and scaling should not be used in the context of representing the geometric equivalent for an assembly structure. There are two subtypes of cartesian_transformation_operator:

- cartesian_transformation_operator_2d
- cartesian_transformation_operator_3d

Attributes

- The ***name*** attribute (inherited from supertype representation_item) provides a naming capability.
- The ***name*** attribute (inherited from functionally_defined_transformation) provides a naming capability.
- The ***description*** attribute can be used to provide optional further textual description.
- The ***axis_1*** attribute the X axis direction of the transformation target.
- The ***axis_2*** attribute is the Y axis direction of the transformation target.
- The ***local_origin*** attribute is the required translation specified as a cartesian point. The actual translation included in the transformation is from the geometric origin to the local origin.
- The ***scale*** attribute is the scaling value specified for the translation.
- The ***axis_3*** attribute is the Z axis direction of the transformation target.

ENTITY cartesian_transformation_operator	Attribute Population	Remarks
representation_item.name	type: label = string	
functionally_defined_transformation .name	type: label = string	
description	type: text = string	OPTIONAL
axis_1	type: entity = direction	OPTIONAL
axis_2	type: entity = direction	OPTIONAL
local_origin	type: entity = cartesian_point	
scale	type: REAL	OPTIONAL
axis	type: entity direction	OPTIONAL Attribute specific to cartesian_transformation _operator_3d

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

A cartesian_transformation_operator_3d defines a geometric transformation composed of translation, rotation, mirroring and uniform scaling. Depending on what kind of transformation is implemented, certain optional attributes are required. With only local_origin given as attribute, the transformation may be an identity or a translation.

2.3.3 Relating shape_representations

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of relating shape representations are illustrated in Diagram 14

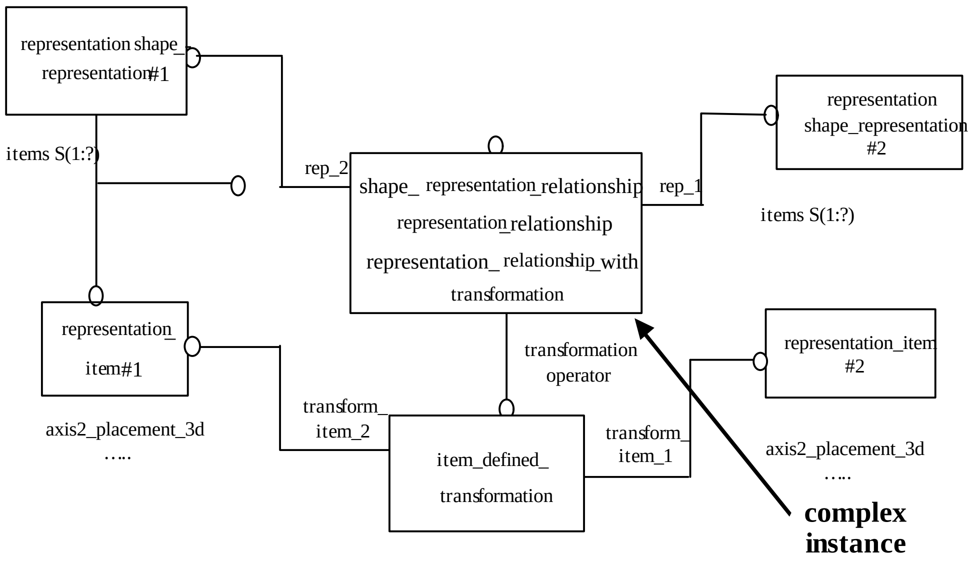

representation shape_representation#1
items S(1:?)
rep_2
shape_representation_relationship
representation_relationship
representation_relationship_with_transformation
rep_1
representation shape_representation #2
items S(1:?)
representation_item#1
axis2_placement_3d
…..
transformation operator
representation_item #2
transform_item_2
item_defined_transformation
transform_item_1
axis2_placement_3d
…..
complex instance

Diagram 14: Relating Shape Representations with Transformation Instance Diagram (item_defined_transformation example)

2.3.3.1 shape_representation_relationship

To define a relationship between two shape_representations that is established via a transformation, a complex instantiation of shape_representation_relationship with representation_relationship_with_transformation is used.

Attributes

- The ***name*** attribute is used to name the relationship.
- The ***description*** attribute can be used to provide optional further textual description.
- The ***rep_1*** attribute references the first of the related shape_representations.
- The ***rep_2*** attribute references the second of the related shape_representations.
- The ***transformation_operator*** attribute specifies the transformation operator that defines the relationship.

ENTITY	Attribute Population	Remarks

shape_representation_relationship		
name	type: label = string	
description	type: text = string	OPTIONAL
rep_1	type: entity = shape_representation	
rep_2	type: entity = shape_representation	
transformation_operator	type: entity item_defined_transformation or functionally_defined_transformation	

Pre-processor Recommendations: the meaning of the rep_1 and rep_2 attributes is specified further by the relation type indicated by the attribute shape_representation_relationship.name.

Post-processor Recommendations: none specific

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#520 = SHAPE_REPRESENTATION('sol1',(#570),#530);
#530 = GEOMETRIC_REPRESENTATION_CONTEXT('c1','external',3);
#540 = CARTESIAN_POINT('', (1.0E+001, 0.0E+000, 1.0E+001));
#550 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#560 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#570 = AXIS2_PLACEMENT_3D('', #540, #550, #560);
#1070 = SHAPE_REPRESENTATION('s_ass',(#1110,#1140),#1080);
#1080 = GEOMETRIC_REPRESENTATION_CONTEXT('ca','',3);
#1110 = AXIS2_PLACEMENT_3D('', #1090, #1095, #1100);
#1120 = CARTESIAN_POINT('', (1.0E+001, 1.0E+000, 1.0E+001));
#1125 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#1130 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#1220 =(REPRESENTATION_RELATIONSHIP('','',#520,#1070)
   REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#1230)
   SHAPE_REPRESENTATION_RELATIONSHIP());
#1230 =ITEM_DEFINED_TRANSFORMATION('p1t','',#1110,#570);
```

Example 11: exchange file segment for relating shape representations

2.3.4 Relating portions of shape

By using the above described shape_representation_relationship concept, geometric relations of individual portions of shape defined as shape_aspects can be represented.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of relating portions of shape are illustrated in Diagram 15.

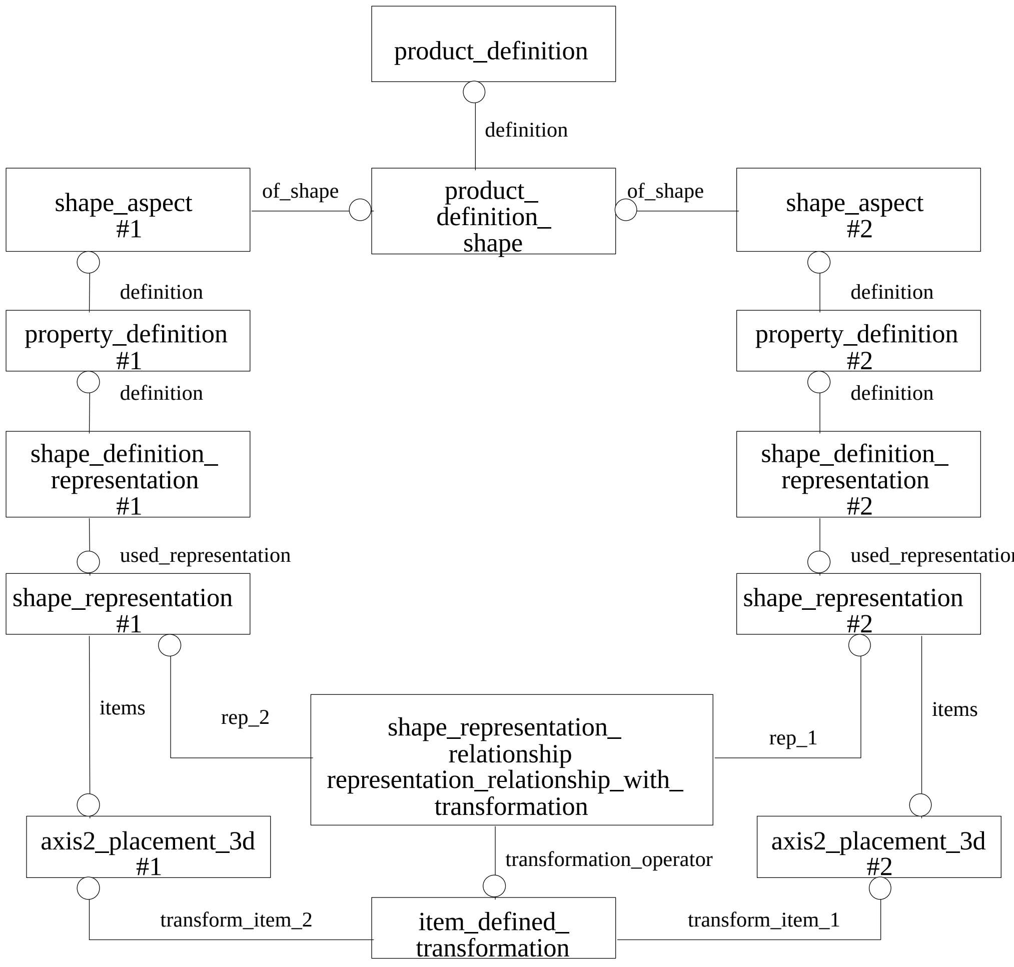


Diagram 15: Specifying Geometric Relations between Portions of Shape Instance Diagram

2.4 Relating part shape properties to product structure

The PDM Schema allows linking geometric structures that result from relating different shape_representations with associated product structure when applicable, i.e. when the geometric structure directly corresponds to the assembly structure. Two alternatives for the implementation of geometric structures related to assembly structures are recommended:

1. The assembly is described with the components built in. With this approach the shape of the component is mapped into the shape of the assembly via mapped_item. The basic idea of the mapped_item is: an item will become part of another item. The assembly component geometry is used as a template in the assembly geometry.
2. The components of an assembly are described together with the construction history. This approach uses the above described representation_relationship_with_transformation. The transformation describes the relation between different workspaces.

The usage of both alternatives is considered reasonable, as both mechanisms make sense even in mixed combinations. With regard to the transformations in the context of assembly a part is in principle incorporated in the assembly only by rigid motion (i.e. translation and/or rotation) excluding mirroring and scaling.

2.5 Explicit Representation of Complete Assembly Geometry

This approach represents the assembled model completely. It is appropriate for explicit representation of assemblies. It uses *mapped_item* in combination with *representation_map* which takes a representation and turns it into an item in a second representation. The transformation to be applied is determined from the mapping origin and the mapping target which are items in the two representations (and therefore founded in the two contexts).

Mapped_item is constrained such that:

- No mapped item shall be dependent on itself to define the representation being mapped (acyclic).
- The mapping origin shall be in the context of the mapping_source representation.

There is no constraint to prevent both representations having the same context (i.e. the new item is shifted in position but does not change spaces). This allows the use of *mapped_item* to position sub-models in a single coordinate space. It is even possible to define an identity matrix transformation by referring to the same entity instance as both mapping_origin and mapping_target. This represents the case where a component is defined in (one of) its final position(s). Until it is mapped onto the second representation, it is not included in the representation even though defined in the same space.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of explicit representation of assembly structure with associated external geometry model structure are illustrated in Diagram 16.

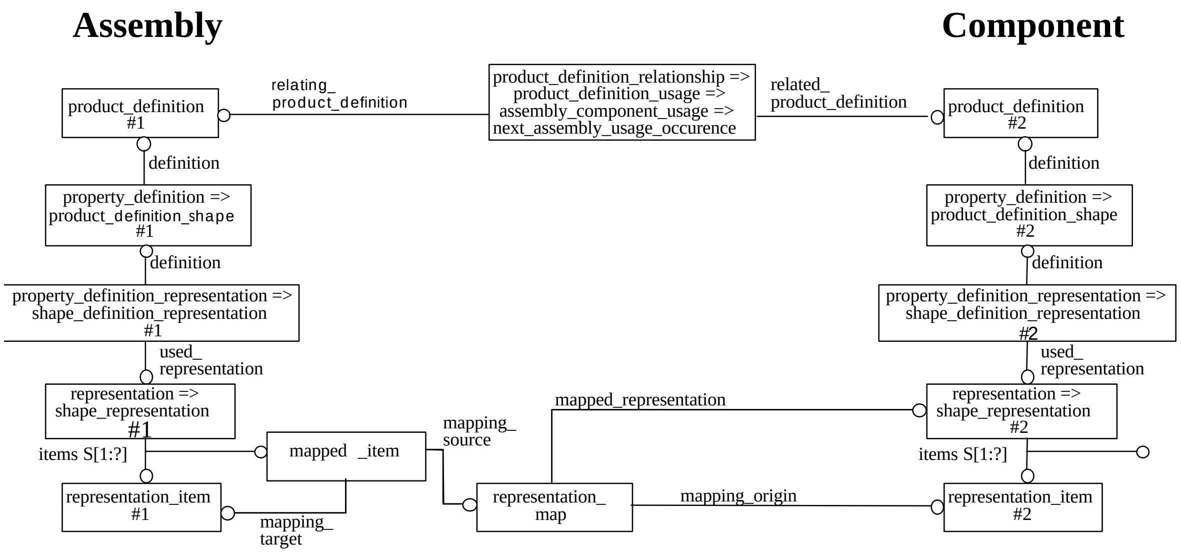

Assembly
Component
product_definition #1
relating_ product_definition
product_definition_relationship => product_definition_usage => assembly_component_usage => next_assembly_usage_occurence
related_ product_definition
product_definition #2
definition
property_definition => product_definition_shape #1
property_definition => product_definition_shape #2
property_definition_representation => shape_definition_representation #1
property_definition_representation => shape_definition_representation #2
used_ representation
representation => shape_representation #1
representation => shape_representation #2
mapped_representation
items S[1:?]
mapped _item
mapping_ source
representation_item #1
mapping_ target
representation_ map
mapping_origin
representation_item #2

Diagram 16: Assembly Structure with Component Templates Instance Diagram

2.5.1.1 mapped_item

The mapped_item allows the use of a representation as a representation_item, e.g., it allows for the definition of representation using other representations. Thus a shape_representation for an assembly component can participate as a template instance in the definition of the shape of the overall assembly.

The representation is defined by mapped_item.mapping_source.mapped_representation. The mapping is achieved through an operator that is a mapping of the mapped_item. mapping_source.mapping_origin onto the mapped_item.mapping_target

Attributes

- The ***mapping_source*** attribute points to an instance of representation_map that specifies the to be mapped representation.
- The ***mapping_target*** attribute is a representation_item (here an axis_placement_3d) that defines the target onto which the representation defined by *mapping_source* is mapped.

ENTITY mapped_item	Attribute Population	Remarks
mapping_source	type: entity = representation_map	
mapping_target	type: entity = representation_item	

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

2.5.1.2 representation_map

Attributes

- The ***mapping_origin*** attribute.
- The ***mapped_representation*** attribute references the to be mapped representation.

ENTITY representation_map	Attribute Population	Remarks
mapping_origin	type: entity = representation_item	should be an axis_placement in the given context
mapped_representation	type: entity = mapped_representation	

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

2.5.2 Implicit Relationships between Assembly Components

This approach represents the components and a set of instructions on how to build the assembly. This approach is adequate for the implicit representation of assemblies. Thus, representation_relationship_with_transformation is used to define the relative positions within an assembly. It does not allow for:

- Inclusion **-** The decision to create a representation which brings all the components together is left to the receiving system.
- The ability to define a component in one position and then replicate it in the same space.

The relationships between the assembly and the component on the product_definition level and the shape_representation level have to be linked through a context_dependent_shape_representation. This is necessary to distinguish between several occurrences of the same component within an assembly.

According to Part 43, the attributes rep_1 and rep_2 of representation_relationship_ with_transformation are determined as:

rep 1 is defined as the representation with a context to which the transformation applies (in other words rep_1 is defined as the representation with a context to which the transformation can be applied in case of building an assembly representation) and

rep 2 is the representation with a context which is the result of the transformation (in other words rep_2 is defined as the representation with the context into which the component representation has to be transformed in case of building an assembly representation).

Based on this definition the attributes of representation_ relationship_with_transformation should be instantiated as follows:

- representation_relationship_with_transformation.rep_1 for the relation to the shape_ representation for the product_definition that is related by assembly_component_usage as related_product_ definition (identifying the component), and
- representation_relationship_with_transformation.rep_2 for the relation to the shape_ representation for the product_definition that is related by assembly_component_usage as relating_ product_definition (identifying the assembly).

Further restrictions are implied by the Part 43 informal proposition on the entity representation_relationship_with_transformation: When the transformation is an item_defined_transformation, the ordering of the representations given for the inherited attributes of representation_relationship shall be consistent with the ordering of the two representation_items given as attributes of item_defined_transformation. Therefore the following usage of the attributes transform_item_1 and transform_item_2 of item_defined_transformation is recommended:

- item_defined_transformation.transform_item_1 for the relation to the representation_ item (e.g. axis2_placement_3d) for the product_definition that is related by assembly_component_usage as related_product_definition (identifying the component) and
- item_defined_transformation.transform_item_2 for the relation to the representation_ item (e.g. axis2_placement_3d) for the product_definition that is related by assembly_component_usage as relating_product_definition (identifying the assembly).

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of relating assemblies and components using item_defined_transformation is illustrated in Diagram 17.

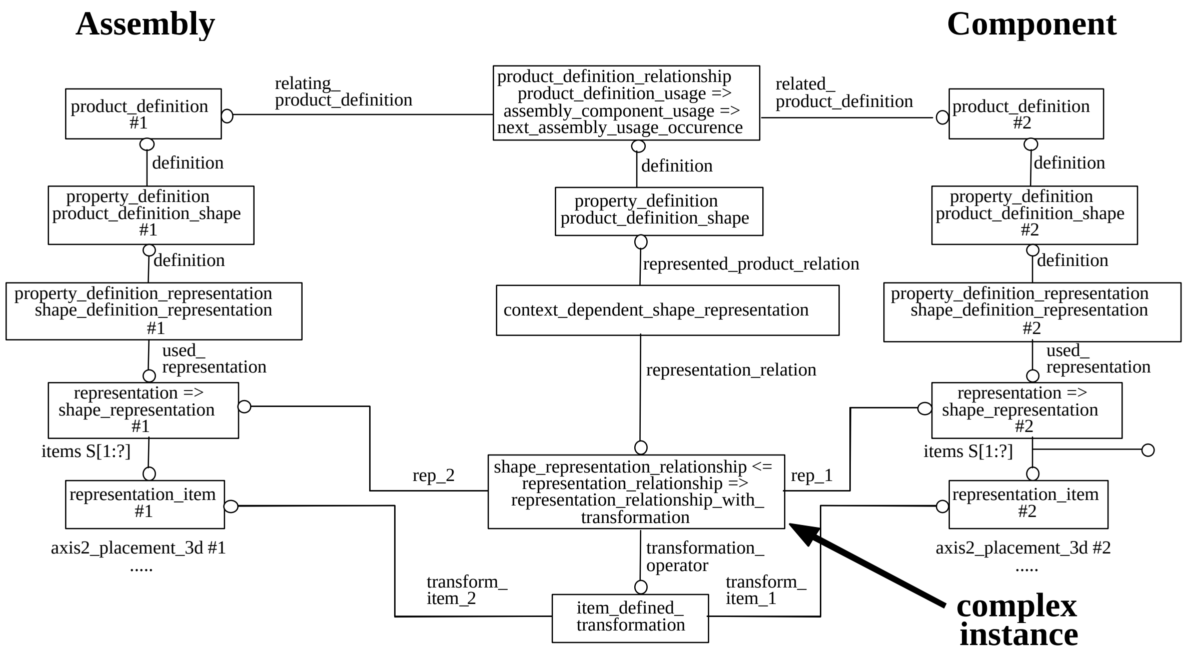

Assembly
Component
product_definition #1
relating_ product_definition
product_definition_relationship product_definition_usage => assembly_component_usage => next_assembly_usage_occurence
related_ product_definition
product_definition #2
definition
property_definition product_definition_shape #1
property_definition product_definition_shape
property_definition product_definition_shape #2
represented_product_relation
property_definition_representation shape_definition_representation #1
context_dependent_shape_representation
property_definition_representation shape_definition_representation #2
used_ representation
representation_relation
representation => shape_representation #1
representation => shape_representation #2
items S[1:?]
rep_2
rep_1
shape_representation_relationship <= representation_relationship => representation_relationship_with_ transformation
representation_item #1
representation_item #2
axis2_placement_3d #1 .....
axis2_placement_3d #2 .....
transformation_ operator
transform_ item_2
transform_ item_1
item_defined_ transformation
complex instance

Diagram 17: Assembly Structure with Defined Component Relationships (usage of item_defined_transformation)

Another possibility is the definition of the assembly construction history via a cartesian_ transformation_operator. The Part 42 definition of cartesian_transformation_operator also allows for scaling and mirroring. But scaling and mirroring shall not to be applied in the context of assemblies. Therefore:

- the scaling value specified by the attribute cartesian_transformation_operator.scale shall be empty (due to being optional; default scaling is equal 1.0) or shall be 1.0, and
- the determinant of the transformation matrix (computed from the attributes axis_1, axis_2 and axis_3 in case of using cartesian_transformation_operator_3d by the function base_axis) shall equal 1.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of relating assemblies and components using cartesian_transformation_operator is illustrated in Diagram 18.

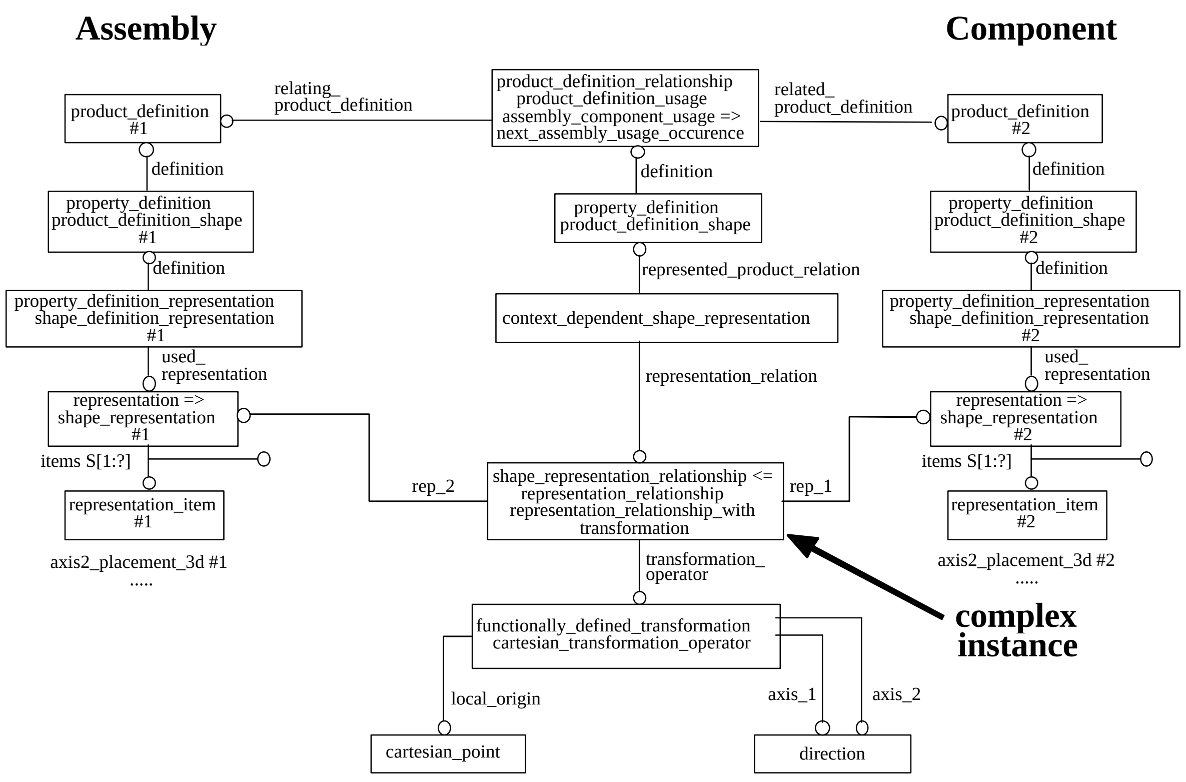

Assembly
Component
product_definition #1
relating_ product_definition
product_definition_relationship product_definition_usage assembly_component_usage => next_assembly_usage_occurence
related_ product_definition
product_definition #2
definition
property_definition product_definition_shape #1
property_definition product_definition_shape
property_definition product_definition_shape #2
represented_product_relation
property_definition_representation shape_definition_representation #1
context_dependent_shape_representation
property_definition_representation shape_definition_representation #2
used_ representation
representation_relation
representation => shape_representation #1
representation => shape_representation #2
items S[1:?]
rep_2
rep_1
shape_representation_relationship <= representation_relationship representation_relationship_with transformation
representation_item #1
representation_item #2
axis2_placement_3d #1 .....
axis2_placement_3d #2 .....
transformation_ operator
functionally_defined_transformation cartesian_transformation_operator
complex instance
local_origin
axis_1
axis_2
cartesian_point
direction

Diagram 18: Assembly Structure with Defined Component Relationships (usage of cartesian_transformation_operator)

2.5.2.1 context_dependent_shape_representation

A context_dependent_shape_representation associates a shape_representation_ relationship with a product_definition_shape. In the given context this allows the explicit specification of a shape of the assembly 'as assembled'. Since elements when assembled might change their shape – e.g. under pressure – this representations may differ from the geometric assembly of the individual shapes.

Attributes

- The ***representation_relation*** attribute is a reference to the shape_representation_relationship used to define the assembly.
- The ***represented_product_relation*** attribute points to the product_definition_shape that is related to the assembly_usage_occurrence.

ENTITY context_dependent_shape_representation	Attribute Population	Remarks
representation_relation	type: entity = shape_representation_relationship	
represented_product_relation	type: entity = product_definition_shape	

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

Complete instantiation example for part properties

The example file contains two independent products. On part is an assembly of two components ('part 1' and part 2'). For this product structure a corresponding relationship that geometrically relates the components with the assembly is defined. It is assumed that the geometry is defined in external files. For 'part 2' this external file constitutes a managed document that is associated to the focused view of the part. For 'part 1' the external geometry is linked in via an external file for which no revision control exists. such a flat file association for external geometry would be typical for files directly received from CAD systems.

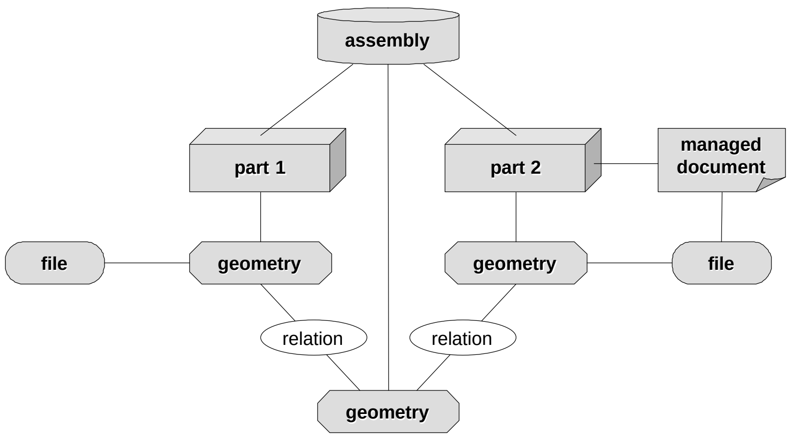


Figure 2 : Schematic Overview of Complete Part Property Example

For the third part in the file, two portions of shape are represented and identified independently. For these portions a geometric relationship is defined. This geometric relationship has no correspondence in product structure.

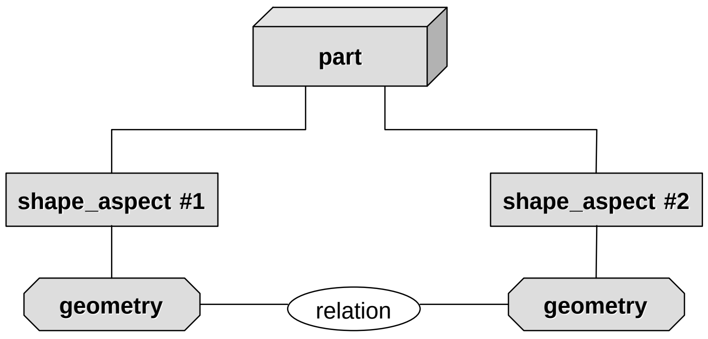


Figure 3 : Schematic Overview of Geometric Relationship

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION((''), '2;1');
FILE_NAME('', '30.06.1999, 12:05:33', ('N:N:'), (''),
   '', '', '');
```

```
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
DATA;

/* Entities #10 to #350 describe meta information and product structure
*/

#10 = APPLICATION_CONTEXT('');
#20 = PRODUCT_CONTEXT('', #10, '');
#30 = PRODUCT('ass', 'assembly', '', (#20));
#40 = PRODUCT_RELATED_PRODUCT_CATEGORY('Assembly', $, (#30));
#50 = PRODUCT_RELATED_PRODUCT_CATEGORY('Part', '', (#30));
#60 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #50, #40);
#70 = PRODUCT('p', 'part', '', (#20));
#80 = PRODUCT('p1', 'part1', '', (#20));
#90 = PRODUCT('p2', 'part2', '', (#20));
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', '', (#30, #80, #90,
#70)
   );
#110 = APPLICATION_PROTOCOL_DEFINITION('version 1.1', 'pdm_schema',
1999
   , #10);
#120 = PRODUCT_DEFINITION_CONTEXT_ROLE('', $);
#130 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, '');
#140 = PRODUCT_DEFINITION_FORMATION('1', '', #30);
#150 = PRODUCT_DEFINITION('assv', 'view on assembly', #140, #130);
#160 = APPLICATION_CONTEXT('mechanical design');
#170 = PRODUCT_DEFINITION_CONTEXT('', #160, 'design');
#180 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#150, #170, #120);
#190 = PRODUCT_RELATED_PRODUCT_CATEGORY('Detail', $, (#80, #90, #70));
#200 = PRODUCT_RELATED_PRODUCT_CATEGORY('Part', '', (#80, #90, #70));
#210 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #200, #190);
#220 = PRODUCT_DEFINITION_FORMATION('1', '', #80);
#230 = PRODUCT_DEFINITION('p1v', 'view on part1', #220, #130);
#240 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#230, $, #120);
#250 = PRODUCT_DEFINITION_FORMATION('1', '', #70);
#260 = PRODUCT_DEFINITION('p', 'view on part', #250, #130);
#270 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#260, $, #120);
#280 = PRODUCT_DEFINITION_FORMATION('1', '', #90);
#290 = PRODUCT_DEFINITION('p2v', 'view on part2', #280, #130);
#300 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#290, $, #120);
#310 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('p1_usage',
   'single instance usage', '', #150, #230, $);
#320 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#330 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#340 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#150, #330, #320);
#350 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('p2_usage',
   'single instance usage', '', #150, #290, $);

/* Entities #370 to #497 define a managed document which */
/* contains a representation of the shape of 'part2'    */
#370 = PRODUCT('p2_shape_doc', 'document reflecting
         shape of part2', '', (#20));
#380 = PRODUCT_RELATED_PRODUCT_CATEGORY('document', '', (#370));
#390 = DOCUMENT_TYPE('configuration controlled document version');
#400 = DOCUMENT('', '', '', #390);
#410 = APPLIED_DOCUMENT_REFERENCE(#290, 'equivalence', (#400));
```

```
#420 = OBJECT_ROLE('mandatory', '');
#430 = ROLE_ASSOCIATION(#420, #410);
#440 = PRODUCT_DEFINITION_FORMATION('1', '', #370);
#450 = DOCUMENT_PRODUCT_EQUIVALENCE('equivalence', '', #400, #440);
#460 = PRODUCT_DEFINITION_CONTEXT('digital document definition', #10,
''
   );
#470 = PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS('p2_shape_d',
   'digital document for part2 shape', #440, #460,(#475));
#475 = DOCUMENT_FILE('p2_shape_file', '', '', #485, ' ', ' ');
#480 = DOCUMENT_REPRESENTATION_TYPE('digital',#475);
#485 = DOCUMENT_TYPE('');
#490 = IDENTIFICATION_ROLE('document_source',$);
#495 = APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT('',#490,#497,(#475));
#497 = EXTERNAL_SOURCE('part1_geometry.stp');


/* Entities #500 to #560 define the geometric model for 'part1' */
#500 = PRODUCT_DEFINITION_SHAPE('shape_p1',$,#230);
#510 = SHAPE_DEFINITION_REPRESENTATION(#500,#520);
#520 = SHAPE_REPRESENTATION('sol1',(#570),#530);
#530 = GEOMETRIC_REPRESENTATION_CONTEXT('c1','external',3);
#540 = CARTESIAN_POINT('', (1.0E+001, 0.0E+000, 1.0E+001));
#550 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#560 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#570 = AXIS2_PLACEMENT_3D('', #540, #550, #560);

/* Entities #575 to #585 state that the shape_representation #520 */
/* is externally defined in a digital file with the location */
/* 'part2_geometry.stp'                                       */
#575 = APPLIED_DOCUMENT_REFERENCE(#230,(#576));
#576 = DOCUMENT_FILE('p1_shape','',$,#577,'','');
#577 = DOCUMENT_TYPE('geometry');
#578 = ROLE_ASSOCIATION(#579,#575);
#579 = OBJECT_ROLE('mandatory',$);
#580 = DOCUMENT_REPRESENTATION_TYPE('digital',#576);
#581 = IDENTIFICATION_ROLE('document_source',$);
#582 = APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT('',#581,#583,(#576));
#583 = EXTERNAL_SOURCE('part2_geometry.stp');
#584 = PROPERTY_DEFINITION('external definition',$,#576);
#585 = PROPERTY_DEFINITION_REPRESENTATION(#584,#520);

/* Entities #600 to #660 define the geometric model for 'part2' */
#600 = PRODUCT_DEFINITION_SHAPE('shape_p2',$,#290);
#610 = SHAPE_DEFINITION_REPRESENTATION(#600,#620);
#620 = SHAPE_REPRESENTATION('sol2',(#670),#630);
#630 = GEOMETRIC_REPRESENTATION_CONTEXT('c2','external',3);
#640 = CARTESIAN_POINT('', (3.0E+001, 0.0E+000, 1.0E+001));
#650 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#660 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#670 = AXIS2_PLACEMENT_3D('', #640, #650, #660);

/* Entities #680 and #690 indicate that the shape_representation #620
*/
/* is externally defined by the file represented by #475
*/
#680 = PROPERTY_DEFINITION('external definition',$,#475);
```

```
#690 = PROPERTY_DEFINITION_REPRESENTATION(#680,#620);


/* Entities #700 to #720 defines the shape and two related portions */
/* for the part 'part' #260                                          */
#700 = PRODUCT_DEFINITION_SHAPE('shape_p',$,#260);
#710 = SHAPE_ASPECT('aspect1',$,#700,.T.);
#720 = SHAPE_ASPECT('aspect2',$,#700,.T.);

/* Entities #730 to #790 define the geometry related to  */
/* shape_aspect #710                                     */
#730 = PROPERTY_DEFINITION('shape for aspect1',$,#710);
#740 = SHAPE_DEFINITION_REPRESENTATION(#730,#750);
#750 = SHAPE_REPRESENTATION('sa1',(#795),#760);
#760 = GEOMETRIC_REPRESENTATION_CONTEXT('ca1','external',3);
#770 = CARTESIAN_POINT('', (3.0E+001, 0.0E+000, 1.0E+001));
#780 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#790 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#795 = AXIS2_PLACEMENT_3D('', #760, #770, #790);
/* the link to the external definition in a file has been omitted here
*/


/* Entities #800 to #860 define the geometry related to  */
/* shape_aspect #720                                     */
#800 = PROPERTY_DEFINITION('shape for aspect2',$,#720);
#810 = SHAPE_DEFINITION_REPRESENTATION(#730,#750);
#820 = SHAPE_REPRESENTATION('sa2',(#790),#760);
#830 = GEOMETRIC_REPRESENTATION_CONTEXT('ca2','external',3);
#840 = CARTESIAN_POINT('', (3.0E+001, 0.0E+000, 1.0E+001));
#850 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#860 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#870 = AXIS2_PLACEMENT_3D('', #840, #850, #860);
/* the link to the external definition in a file has been omitted here
*/

/* #1000 and #1010 establish the relationship between the */
/* geometry of the two shape_aspects                        */
#1000 = (REPRESENTATION_RELATIONSHIP('','',#750,#820)
   REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#1010)
   SHAPE_REPRESENTATION_RELATIONSHIP());
#1010 = ITEM_DEFINED_TRANSFORMATION('aspect_rel','',#800,#870);


/* #1050 to #1140 define the geometric representation of  */
/* the assembly (#150). The shape_representation contains */
/* two axis_placements that are the reference points for  */
/* the transformations of the components                  */
#1050 = PRODUCT_DEFINITION_SHAPE('shape_of_ass',$,#150);
#1060 = SHAPE_DEFINITION_REPRESENTATION(#1050,#1070);
#1070 = SHAPE_REPRESENTATION('s_ass',(#1110,#1140),#1080);
#1080 = GEOMETRIC_REPRESENTATION_CONTEXT('ca','',3);
#1090 = CARTESIAN_POINT('', (3.0E+001, 4.0E+000, 1.0E+001));
#1095 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#1100 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#1110 = AXIS2_PLACEMENT_3D('', #1090, #1095, #1100);
#1120 = CARTESIAN_POINT('', (1.0E+001, 1.0E+000, 1.0E+001));
```

```
#1125 = DIRECTION('', (1.0E+000, 0.0E+000, 0.0E+000));
#1130 = DIRECTION('', (0.0E+000, -1.0E+000, 0.0E+000));
#1140 = AXIS2_PLACEMENT_3D('', #1120, #1125, #1130);

/* Entities #1200 to #1230 define the geometric relationship */
/* between the component 'part1' and the assembly #150. This */
/* relationship is linked to the product structure relationship */
/* defined by the next_assembly_usage_occurrence #310         */
#1200 = PRODUCT_DEFINITION_SHAPE('shape of assembled p1',$,#310);
#1210 = CONTEXT_DEPENDENT_SHAPE_REPRESENTATION(#1200,#1220);
#1220 =(REPRESENTATION_RELATIONSHIP('','',#520,#1070)
   REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#1230)
   SHAPE_REPRESENTATION_RELATIONSHIP());
#1230 =ITEM_DEFINED_TRANSFORMATION('p1t','',#1110,#570);


/* Entities #1300 to #1330 define the geometric relationship */
/* between the component 'part2' and the assembly #150. This */
/* relationship is linked to the product structure relationship */
/* defined by the next_assembly_usage_occurrence #350         */
#1300 = PRODUCT_DEFINITION_SHAPE('shape of assembled p2',$,#350);
#1310 = CONTEXT_DEPENDENT_SHAPE_REPRESENTATION(#1300,#1320);
#1320 =(REPRESENTATION_RELATIONSHIP('','',#620,#1070)
   REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#1330)
   SHAPE_REPRESENTATION_RELATIONSHIP());
#1330 =ITEM_DEFINED_TRANSFORMATION('p2t','',#1140,#670);


/* Entities #5000 to #5070 assert that the in #290 defined view */
/* of part2 has a mass of 3KG                                   */
#5000 = GENERAL_PROPERTY('', 'mass property', '');
#5010 = GENERAL_PROPERTY_ASSOCIATION('non-definitional', '', #5000,
#5020);
#5020 = PROPERTY_DEFINITION('', '', #290);
#5030 = PROPERTY_DEFINITION_REPRESENTATION(#5020, #5040);
#5040 = REPRESENTATION('property value', (#5060), #5050);
#5050 = REPRESENTATION_CONTEXT('', '');
#5060 = MEASURE_REPRESENTATION_ITEM('mass', MASS_MEASURE(3000), #5070);
#5070 =(NAMED_UNIT(*) MASS_UNIT() SI_UNIT($,.GRAM.));

ENDSEC;
END-ISO-10303-21;
```

2.6 Known issues

2.6.1 Identification of geometric models

This document recommends the use of the attribute name of shape_representation to define an identification of a geometric model. This practice is in alignment with current CAx implementations and the descriptions in Part 43 that define representation.name as an identifier for the representation. AP214 does not map the identification to shape_representation.name but rather uses an instance of id_attribute that is associated to the shape_representations. A ballot comment to AP214 DIS to adapt the approach recommended herein will be submitted.

2.6.2 Material properties

Some application protocols such as AP214 use specific subtypes to specify material properties. These subtypes are currently not included in the PDM Schema.

2.6.3 Mapping of part properties in AP 214

The requirement to represent properties for parts is in AP214 always mapped onto an AIM structure that extends the property_definition tree with a general_property. In consequence AP214 compliant implementations are forced to always instantiate general_property when part properties shall be represented. We feel that this is not adequate, since PDM systems in general do not have the capability to distinguish between properties related to parts and properties as a general characterization criteria independent from the actually present product data population. A corresponding AP214 DIS ballot comment will be submitted.

3 Part Structure and Relationships

The STEP PDM Schema supports explicit hierarchical product structures representing assemblies and the constituents of those assemblies. This explicit part structure corresponds to the traditional engineering and manufacturing bill of material indentured parts list. Relationships between part view definitions are the principle elements used to structure an explicit part assembly configuration. The relationship itself represents a specific *usage occurrence* of the constituent part definition within the immediate parent assembly definition. Mechanisms to represent quantity associated with this assembly-component usage relationship are also provided.

The PDM Schema also has the capability to identify and track a specified usage of a component definition in an assembly at a higher level than the immediate parent subassembly. Consider a wheel-axle subassembly composed of one axle and two wheels, the right and the left. A higher-level chassis assembly is in turn composed of two wheel-axle subassemblies, the front and the rear. The requirement to individually identify the left-front wheel, for example, is supported by this capability.

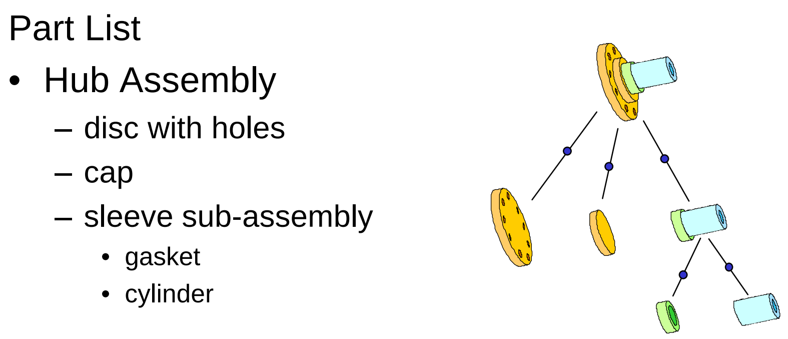
Different view definitions of the same version of a part may participate in different explicit product structures. For example, a design/as-planned view of a particular version of a part, representing the design discipline part definition, may be engaged in an explicit design assembly structure. A manufacturing/as-built view of the same part version represents the definitional template for the actual physical part, and may participate in a manufactured assembly structure that is different from the design assembly structure. Finally, a support/as-maintained view of the part version representing the physical part definition may participate in yet another different disassembly structure.

In addition to hierarchical assembly structures, the STEP PDM Schema supports relationships between parts to characterize explicit alternates and substitutes for the assembly. Other relationships between part definitions exist to characterize the make from relationship and for supplied part identification.

3.1 Explicit Assembly Bill Of Material

The STEP PDM Schema supports explicit hierarchical assembly structures representing assemblies and the constituents of those assemblies. A constituent of an assembly may itself also be an assembly (a subassembly), which in turn is made up of constituents. Detail components make up the leaves of these hierarchical part structures. This explicit part structure corresponds to the traditional engineering and manufacturing bill of material parts list.

Each individual node in this hierarchy is represented by a part master identification. The relationships are made between product_definition entities representing a view definition of the part master. The relationship itself represents the usage occurrence of a constituent definition within the immediate parent assembly definition. In the simplest form, this structure corresponds to a basic indentured parts list (see Figure 4).



Part List

- Hub Assembly
  - – disc with holes
  - – cap
  - – sleeve sub-assembly
    - gasket
    - cylinder

Figure 4 : Basic Indentured Part List

Each node in the assembly structure breakdown is a part master identification, representing the definitional template for the part. The nodes in the product structure each have a corresponding entry in the part list, with an indenture level corresponding to depth in the structure.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of basic explicit assembly structure are shown in Diagram 19.

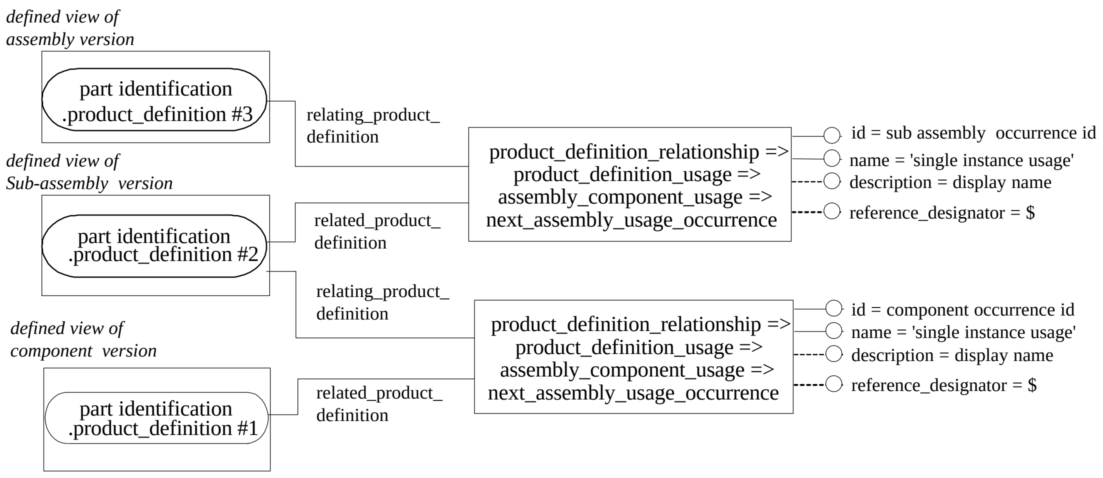

defined view of
assembly version
part identification
.product_definition #3
relating_product_
definition
defined view of
Sub-assembly version
part identification
.product_definition #2
related_product_
definition
product_definition_relationship =>
product_definition_usage =>
assembly_component_usage =>
next_assembly_usage_occurrence
id = sub assembly occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
relating_product_
definition
defined view of
component version
part identification
.product_definition #1
related_product_
definition
product_definition_relationship =>
product_definition_usage =>
assembly_component_usage =>
next_assembly_usage_occurrence
id = component occurrence id
name = 'single instance usage'
description = display name
reference_designator = $

Diagram 19 : Explicit Assembly Instance Diagram

3.1.1.1 assembly_component_usage

This entity represents the general relationship between two part master identifications, one a definition of a component and the other a definition of the parent assembly. This entity is normally not recommended to be instantiated as itself, but is typically instantiated as the subtype next_assembly_usage_occurrence. This subtype represents a unique individual occurrence of the component definition as used within the parent assembly. The assembly_component_usage is only instantiated as itself if there is an explicit requirement to represent the general relationship between a component and an assembly definition.

Attributes

- The ***id*** attribute provides a unique identifier for the relationship.
- The ***name*** attribute provides a general nomenclature for the relationship.
- The ***description*** attribute allows for additional text that further may describe the relationship.
- The ***relating_product_definition*** attribute is a reference to the parent assembly.
- The ***related_product_definition*** attribute is a reference to the constituent of the assembly.
- The ***reference_designator*** attribute is optional.

ENTITY assembly_component_usage	Attribute Population	Remarks
id	type : identifier = string	Should be unique in combination with a given relating (parent) and related (child) product_definition.
name	type : label = string	

description	type : text = string	OPTIONAL
relating_product_definition	type : entity = product_definition	reference to assembly
related_product_definition	type : entity = product_definition	reference to component
reference_designator	type : identifier = string	OPTIONAL

Pre-processor Recommendations: No cyclic product_definiton_relationship structures may be defined. The value of the id attribute must be unique in combination with the same relating assembly and related component definitions. It is generally recommended that only subtypes of assembly_component_usage be instantiated. It is not recommended to instantiate the optional reference_designator attribute for an instance of assembly_component_usage.

Post-processor Recommendations: When importing an instance of assembly_component_usage as itself, postprocessors should interpret this as representing the general assembly-component relationship.

Related Entities: none specific

3.1.1.2 next_assembly_usage_occurrence

This entity is a subtype of assembly_component_usage. It represents a single individual occurrence of a component definition as used in an immediate next higher parent assembly. The id attribute contains a unique instance identifier for the individual component occurrence.

Pre-processor Recommendations: No cyclic product_definiton_relationship structures may be defined. The value of the id attribute must be unique between the same assembly and component definitions.
The name attribute should contain the value 'single instance usage' to identify this as a single occurrence of the constituent definition used in the immediate parent assembly. It is recommended the description attribute contain an instance display name label for the usage occurrence, if one exists. The reference designator attribute is optional: it may be used to represent a positional designation for the component within the assembly.

Post-processor Recommendations: When importing an instance of next_assembly_usage_occurrence as itself, postprocessors should interpret this to represent a single individual usage occurrence of the component definition within the parent assembly definition.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context and life cycle stage */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');

/* optional AP214 specific characterization for assembly definition */
#340 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#350 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#520 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #350, #340);

/* type discriminator for part as product */
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#380, #440));

/* part master for component definition */
#360 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#380));
#380 = PRODUCT('g1', 'gasket', $, (#220));
```

```
#390 = PRODUCT_DEFINITION_FORMATION('A','',#380);
#400 = PRODUCT_DEFINITION('gv1', 'design view on gasket', #390, #230);

/* part master for assemly definition */
#420 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', (#440));
#440 = PRODUCT('s1', 'sleeve assembly', $, (#220));
#450 = PRODUCT_DEFINITION_FORMATION('A','',#440);
#460 = PRODUCT_DEFINITION('sv1', 'design view on sleeve assembly',
#450, #230);

/* single usage occurrence of component defn within assembly defn */
#530 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu1', 'single instance usage',
   'gasket usage 1', #460, #400, $);
```

Example 12 : exchange file for explicit assembly

3.1.2 Quantified Component Usage

The basic explicit assembly represents a single occurrence of the component definition within the assembly definition. Multiple occurrences of a constituent used in an assembly are represented in two ways in the PDM Schema:

- quantified usage occurrence
- multiple individual usage occurrences

The quantified usage occurrence associates a quantity with the component usage, but does not allow independent identification of individual occurrences when a component definition is used multiple times.

The PDM Schema uses separate single usage occurrences to represent multiple occurrences of a component within an assembly when the various components must be distinguished individually (see 3.1.3).

The quantified usage occurrence associates a quantity value with the multiple use of a component definition in the parent assembly definition. The quantified usage occurrence simply relates a quantity value with the multiple component occurrences, it does not distinguish the individual occurrences independently. The quantified usage occurrence is most commonly used for standard component parts, such as fasteners, that need to be quantified but do not need to be individually distinguished. This structure corresponds to a basic indentured parts list with the associated part quantity values (see Figure 5).

Part List	Quantity
• Hub Assembly	
– disc with holes	1
– cap	1
– sleeve sub-assembly	1
• cylinder	1
• gasket	2

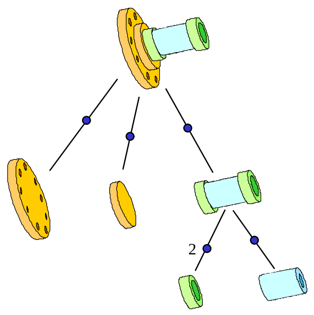


Figure 5 : Indentured Parts List with Associated Quantity

As with the basic indentured parts list, each node in the part structure breakdown is a part master, representing the definitional template for the component or assembly. The nodes of the product structure each have a corresponding entry in the parts list, with the indenture level corresponding to depth in the assembly structure. The quantity of each usage occurrence is identified in the parts list - the single occurrence usage has an implicit quantity of one. In this example, the relationship between the gasket component definition and the parent sleeve subassembly is represented by a quantified component usage to denote two occurrences of the gasket definition used in the parent sleeve subassembly.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements for quantified assembly structure are shown in Diagram 20.

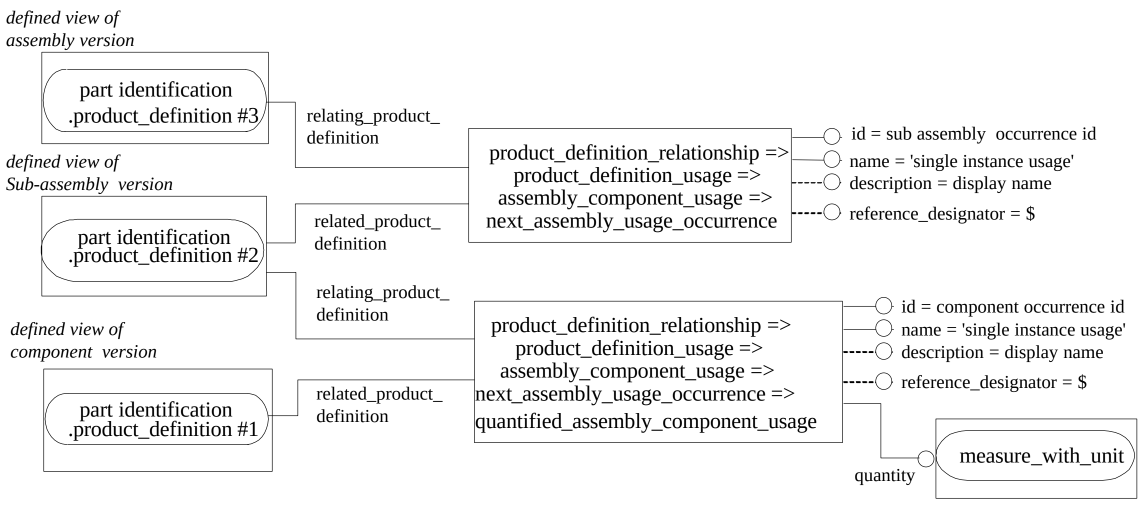

defined view of assembly version
part identification .product_definition #3
relating_product_definition
defined view of Sub-assembly version
part identification .product_definition #2
related_product_definition
product_definition_relationship => product_definition_usage => assembly_component_usage => next_assembly_usage_occurrence
id = sub assembly occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
relating_product_definition
defined view of component version
part identification .product_definition #1
related_product_definition
product_definition_relationship => product_definition_usage => assembly_component_usage => next_assembly_usage_occurrence => quantified_assembly_component_usage
id = component occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
quantity
measure_with_unit

Diagram 20 : Quantified Assembly Instance Diagram

3.1.2.1 quantified_assembly_component_usage

This entity is a subtype of assembly_component_usage. When the quantified_assembly_component_usage is required, it should be created as an AND instantiation with the next_assembly_usage_occurrence. It adds the attribute quantity to identify the number of occurrences of the constituent definition that are used in the parent assembly. With the quantified_assembly_component_usage, multiple individual occurrences of the constituent part are not distinguished independently.

Attributes
- The ***quantity*** attribute gives the number of usage occurrences of a component within an assembly.

ENTITY quantified_assembly_component _usage	Attribute Population	Remarks
id	see supertype	
name	see supertype	
description	see supertype	OPTIONAL
relating_product_definition	see supertype	reference to assembly
related_product_definition	see supertype	reference to component
reference_designator	see supertype	OPTIONAL should not be instantiated
quantity	type : entity = measure_with_unit	

Pre-processor Recommendations: This entity should be instantiated when quantities need to be represented, but when the individual occurrences do not need to be independently distinguished. This is created as a "complex" instance of next_assembly_usage_occurrence AND quantified_assembly_component_usage. For a quantity of one, a simple instance of next_assembly_usage_occurrence alone may be used. However, quantified_assembly_component_usage needs to be used if the number of occurrences is specified by a specific unit, e.g. the usage of "one liter of oil" as a component in an assembly.

Post-processor Recommendations: When importing an instance of next_assembly_usage_occurrence AND quantified_assembly_component_usage, a quantity value of one should be interpreted as equivalent to the simple next_assembly_usage_occurrence.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context and life cycle stage */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');


/* type discriminator for part as product */
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#380, #440));

/* part master for component definition */
#360 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#380));
#380 = PRODUCT('g1', 'gasket', $, (#220));
#390 = PRODUCT_DEFINITION_FORMATION('A','',#380);
#400 = PRODUCT_DEFINITION('gv1', 'design view on gasket', #390, #230);

/* part master for assemly definition */
#420 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $,(#440));
#440 = PRODUCT('s1', 'sleeve assembly', $, (#220));
#450 = PRODUCT_DEFINITION_FORMATION('A','',#440);
#460 = PRODUCT_DEFINITION('sv1', 'design view on sleeve assembly',
#450, #230);

/* quantified multiple use occurrence of component within assembly */
#860 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#870 = NAMED_UNIT(#860);
#880 = MEASURE_WITH_UNIT(COUNT_MEASURE(2), #870);
#890 = (ASSEMBLY_COMPONENT_USAGE($) NEXT_ASSEMBLY_USAGE_OCCURRENCE()
   PRODUCT_DEFINITION_RELATIONSHIP('guQ', 'quantified instance usage',
   'quantified usage of gasket', #460, #400) PRODUCT_DEFINITION_USAGE()
   QUANTIFIED_ASSEMBLY_COMPONENT_USAGE(#880));
```

Example 13: exchange file for quantified component usage

3.1.3 Multiple Individual Component Occurrences

The basic explicit assembly represents a single occurrence of the component definition within the assembly definition. Multiple occurrences of a constituent used within an assembly are represented in one of two ways in the PDM Schema:

- quantified usage occurrence
- multiple individual usage occurrences

The quantified usage occurrence identifies the component quantity, but does not allow independent identification of the individual occurrences when a component definition is used multiple times (see 3.1.2).

The PDM Schema uses separate single usage occurrences to represent multiple occurrences of a component within an assembly when the various components must be distinguished individually. Multiple occurrences of a component within an assembly must be individually identified to allow separate ids, display names, reference designation, or related property information, including shape and orientation/transformation information.

For example, consider a component that is a flexible pipe. The geometrical representation assigned to the definition of that component may be a simple I shaped tube. The geometrical representation associated with one occurrence of the pipe positioned in a parent assembly may be an S shape, as constrained by the assembly environment. A second occurrence of the pipe definition may be used in the same parent assembly, but in a different position. The geometrical representation assigned to this occurrence may be an L shape due to different constraints from the different position of the component within the parent assembly.

This structure corresponds to a basic indentured parts list with the associated part quantity values where the individual component occurrences are individually identified.

Part List	Quantity
• Hub Assembly	
– disc with holes d-1	1
– cap ca-1	1
– sleeve sub-assembly cg-1	1
• cylinder c-1	1
• gasket gu-1 gu-2	2

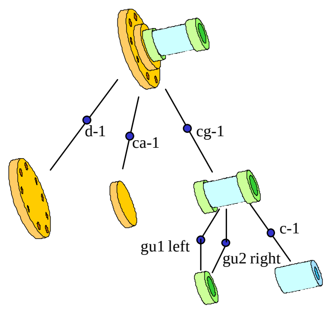


Figure 6 : Indentured Parts List with Multiple Individual Components

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of individually distinguished occurrences of a constituent in an assembly structure are illustrated in Diagram 21.

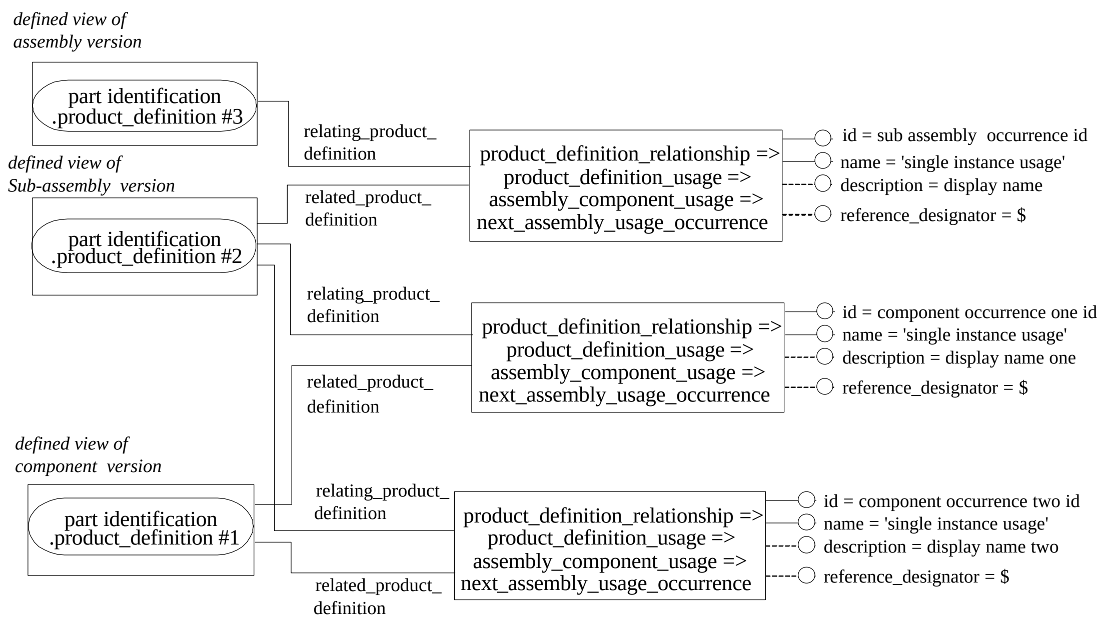

defined view of
assembly version
part identification
.product_definition #3
defined view of
Sub-assembly version
part identification
.product_definition #2
defined view of
component version
part identification
.product_definition #1
relating_product_
definition
related_product_
definition
product_definition_relationship =>
product_definition_usage =>
assembly_component_usage =>
next_assembly_usage_occurrence
id = sub assembly occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
id = component occurrence one id
name = 'single instance usage'
description = display name one
reference_designator = $
id = component occurrence two id
name = 'single instance usage'
description = display name two
reference_designator = $

Diagram 21 : Multiple Individual Component Usage Instance Diagram

Pre-processor Recommendations: The entity next_assembly_usage_occurrence (see 3.1.1.2) should be instantiated once for each individual occurrence of a component definition used in the parent assembly. The value of the attribute id must be unique for each instance. The description attribute may contain a unique display name for presentation of the assembly structure.

Post-processor Recommendations: The sum of single instances of next_assembly_usage_occurrence that exist between an assembly definition and a component definition represents the total quantity of component occurrences used in the parent assembly.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context and life cycle stage */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');

/* optional AP214 specific characterization for assembly definition */
#340 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#350 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#520 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #350, #340);
```

```
/* type discriminator for part as product */
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#380, #440));

/* part master for component definition */
#360 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#380));
#380 = PRODUCT('g1', 'gasket', $, (#220));
#390 = PRODUCT_DEFINITION_FORMATION('A','',#380);
#400 = PRODUCT_DEFINITION('gv1', 'design view on gasket', #390, #230);

/* part master for assemly definition */
#420 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $, (#440));
#440 = PRODUCT('s1', 'sleeve assembly', $, (#220));
#450 = PRODUCT_DEFINITION_FORMATION('A','',#440);
#460 = PRODUCT_DEFINITION('sv1', 'design view on sleeve assembly',
#450, #230);

/* multiple single occurrences of component defn in assembly defn */
#530 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu-1', 'single instance usage',
   'gu1 left', #460, #400, $);
#540 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu-2', 'single instance usage',
   'gu2 right', #460, #400, $);
```

Example 14: exchange file for multiple individual component usages

3.1.4 Promissory Component Usage

The STEP PDM Schema supports promissory usage relationships between components and some higher level assembly that is not the immediate parent in the hierarchy. This may be important to relate an outsourced component to a top-level assembly definition related to a configuration and product_concept. A supplier may only require the configuration and end item identification that is associated with the component of interest, and may not need to know the detailed assembly structure in between. The promissory component usage may be useful during early design phases to create preliminary BOMs for prototyping.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of promissory assembly are illustrated in Diagram 22.

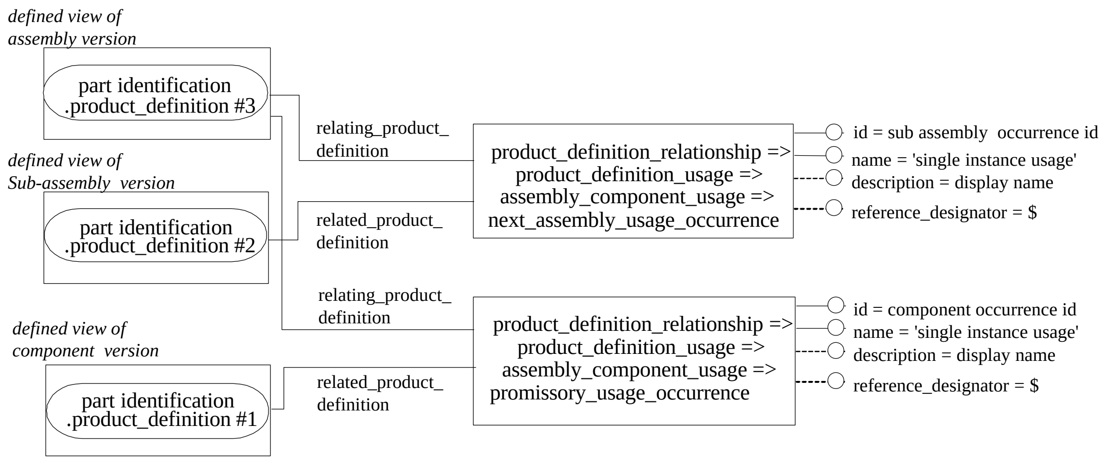

defined view of
assembly version
part identification
.product_definition #3
defined view of
Sub-assembly version
part identification
.product_definition #2
defined view of
component version
part identification
.product_definition #1
relating_product_
definition
related_product_
definition
relating_product_
definition
related_product_
definition
product_definition_relationship =>
product_definition_usage =>
assembly_component_usage =>
next_assembly_usage_occurrence
product_definition_relationship =>
product_definition_usage =>
assembly_component_usage =>
promissory_usage_occurrence
id = sub assembly occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
id = component occurrence id
name = 'single instance usage'
description = display name
reference_designator = $

Diagram 22: Promissory Assembly Usage Instance Diagram

3.1.4.1 promissory_usage_occurrence

This entity is a subtype of assembly_component_usage. It represents the usage occurrence of a component within a higher level assembly that is not the immediate parent, in the case where the detailed assembly structure in between the component and the higher level assembly is not represented. A promissory_usage_occurrence specifies the intention to use the constituent in an assembly. It may also be used when the product structure is not completely defined.

Pre-processor Recommendations: The higher level assembly should be associated with a configuration item with its related end item identification in the typical usage case for the promissory_usage_occurrence. If the assembly structure between the component and the higher level assembly must be specified, the next_assembly_usage_occurrence should be used (see 3.1.1.2).

Post-processor Recommendations: none specific

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context and life cycle stage */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10,'');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');
```

```
/* optional AP214 specific characterization for assembly definition */
#340 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#350 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#520 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #350, #340);

/* type discriminator for part as product */
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#380, #440));

/* part master for component definition */
#360 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#380));
#380 = PRODUCT('g1', 'gasket', $, (#220));
#390 = PRODUCT_DEFINITION_FORMATION('A','',#380);
#400 = PRODUCT_DEFINITION('gv1', 'design view on gasket', #390, #230);

/* part master for top level assemly definition */
#420 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $, (#440));
#440 = PRODUCT('h1', 'hub assembly', $, (#220));
#450 = PRODUCT_DEFINITION_FORMATION('A','',#440);
#460 = PRODUCT_DEFINITION('hv1', 'design view on hub assembly', #450,
#230);

/* promissory usage occurrence of component in top level assembly */
#530 = PROMISSORY_USAGE_OCCURRENCE('pu-1', 'promissory instance usage',
'pu-1 promissory', #460, #400, $);
```

Example 15: exchange file for promissory assembly usage

3.2 Multi-Level Assembly Digital Mock Up

STEP has the capability to identify individual occurrences of component in a multi-level assembly (see 3.1.3). This provides the ability to assign to each occurrence an identifier, a position in the assembly, a geometrical representation, or other properties that may be different from that assigned to the part definition of the component.

In order to distinguish a specific occurrence of a component in an assembly of more than two hierarchical levels, the specified_higher_usage_occurrence entity is used. For example, a wheel-axle subassembly is composed of an axle and two wheels, the right and the left. A higher level chassis assembly is in turn composed of two wheel-axle subassemblies, the front and the back. The requirement to individually identify the left-front wheel, for example, is supported by this capability.

From the previous wheel-axle-chassis example, an instance of specified_higher_usage_occurrence is used to represent the left-front wheel. The related_product_definition attribute references the product_definition that represents the part definition of the component wheel. The relating_product_definition attribute references the product_definition representing the definition of the higher level chassis assembly. The attribute next_usage is a reference to the next_assembly_usage_occurrence that represents the left wheel occurrence in the wheel-axle subassembly. The attribute upper_usage is a reference to the next_assembly_usage_occurrence representing the front occurrence of the wheel-axle subassembly within the higher level chassis assembly.

This structure corresponds to an indentured parts list with associated part quantity values and individual identification of the specific component occurrences used within a multi-level assembly (see Figure 7).

Part List	Quantity
• Hub Assembly	
– disc with holes	1
d-1	
– cap	1
ca-1	
– sleeve sub-assembly	2
su-1 *(rear)*	
• cylinder	1
cu-1 *(rear)*	
• gasket	2
gu-1a *(rear left)*	
gu-2a *(rear right)*	
su-2 *(front)*	
• cylinder	1
cu-2 *(front)*	
• gasket	2
gu-1b *(front left)*	
gu-2b *(front right)*	

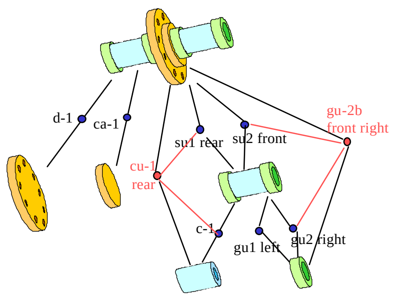


Figure 7 : Indentured Parts List with Specified Higher Component Usage

In Figure 7, the specific higher usage cu-1 of the cylinder (c-1 in a sleeve subassembly) within the rear sleeve (su-1 in a hub assembly) of the higher level hub assembly is illustrated. Also illustrated is the right gasket (gu2 within a sleeve subassembly) within the front sleeve (su-2 within a hub assembly) of the higher level hub assembly, individually identified as gu-2b. Each specified higher usage occurrence of the gasket, as well as that of the cylinder, would be individually identified using the same mechanism as illustrated for the rear cylinder and the front right gasket.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of explicit multi-level assemblies and digital mockup are illustrated in Diagram 23.

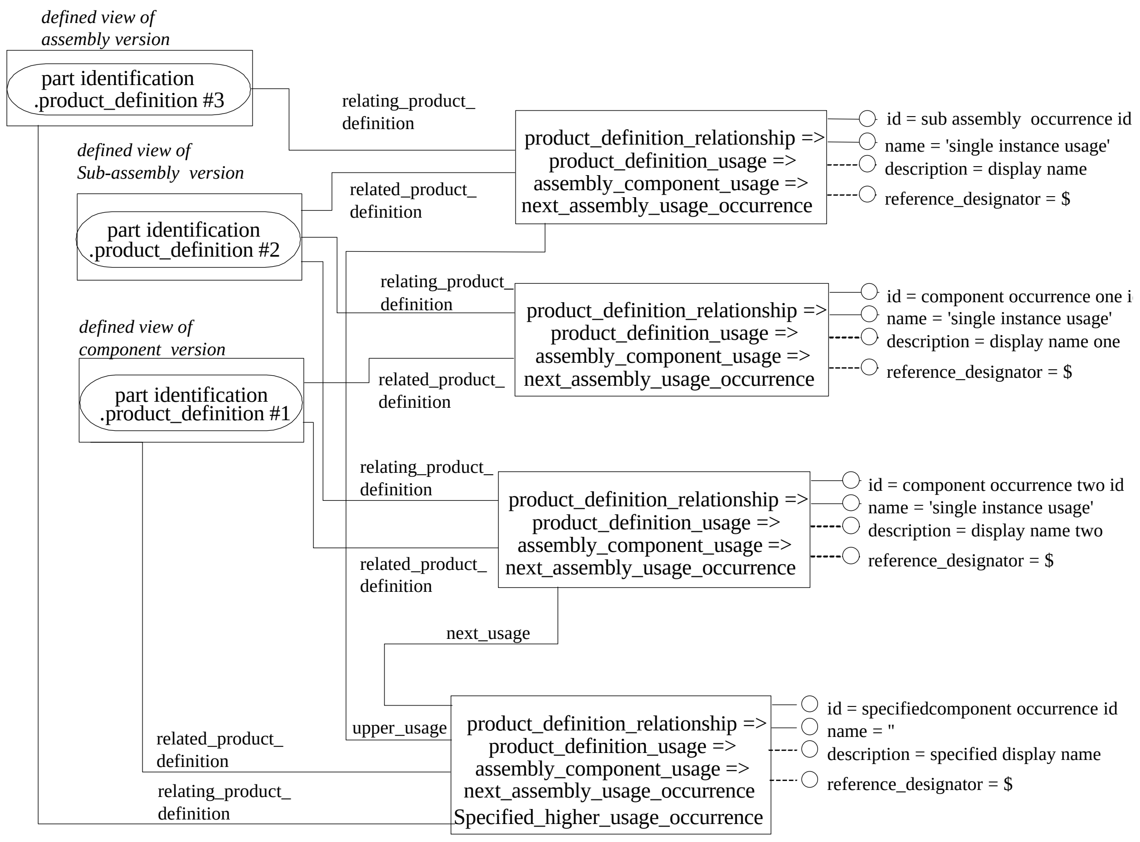

defined view of assembly version
part identification .product_definition #3
defined view of Sub-assembly version
part identification .product_definition #2
defined view of component version
part identification .product_definition #1
relating_product_ definition
related_product_ definition
product_definition_relationship => product_definition_usage => assembly_component_usage => next_assembly_usage_occurrence
id = sub assembly occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
id = component occurrence one i
name = 'single instance usage'
description = display name one
reference_designator = $
id = component occurrence two id
name = 'single instance usage'
description = display name two
reference_designator = $
next_usage
upper_usage
product_definition_relationship => product_definition_usage => assembly_component_usage => next_assembly_usage_occurrence Specified_higher_usage_occurrence
id = specifiedcomponent occurrence id
name = ''
description = specified display name
reference_designator = $

Diagram 23 : Multi-level Assembly DMU Instance Diagram

3.2.1.1 specified_higher_usage_occurrence

This entity is a subtype of assembly_component_usage. This entity represents the specific use occurrence within a higher level assembly of an individual occurrence of a component definition used in an immediate parent sub-assembly.

EXAMPLE - a wheel-axle subassembly is composed of an axle and two wheels, the right and the left. A higher level chassis assembly is in turn composed of two wheel-axle subassemblies, the front and the back. One specific higher usage of the left wheel occurrence within the higher level chassis assembly is in the front wheel-axle subassembly (other is in the rear). The specified_higher_usage_occurrence entity represents the individually identified left-front wheel. A second instance of this entity could represent another specified higher usage occurrence of the wheel, for example left-rear or right-front.

In addition to inherited attributes, this entity defines the attributes next_usage and upper_usage to relate a single usage occurrence at one level in the assembly hierarchy (the next_usage) to a single usage occurrence at the next higher level in the assembly (the upper usage).

In the case of a specified usage occurrence at more than one higher level in the assembly hierarchy, the upper_usage attribute will reference the specified_higher_usage instance representing the specified occurrence at the higher level. In this way, instances of specified_higher_usage_occurrence are "chained" together to identify specific occurrences of components in an assembly hierarchy with an arbitrary number of levels.

Attributes

- The ***next_usage*** attribute identifies the single usage occurrence at a given level in the hierarchy.
- The ***upper_usage*** attribute identifies the specific higher usage occurrence one level up in the hierarchy.

ENTITY specified_higher_usage_occurrence	Attribute Population	Remarks
id	see supertype	
name	see supertype	
description	see supertype	OPTIONAL
relating_product_definition	see supertype	
related_product_definition	see supertype	
reference_designator	see supertype	OPTIONAL
next_usage	type : entity = next_assembly_usage_occurrence	
upper_usage	type : entity = assembly_component_usage (next_assembly_usage_occurrence OR specified_higher_usage_occurrence)	References s_h_u_o in the case of a specified usage at more than one higher level in the hierarchy

Pre-processor Recommendations: This entity should be instantiated once for each specific use in a higher level assembly of an individual occurrence of a component definition used in a parent assembly. The value of the attribute id must be unique for each instance. The description attribute may contain a unique display name for presentation of the assembly structure

Post-processor Recommendations: none specific

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context and life cycle stage */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');

/* optional AP214 specific characterization for assembly definitions */
#340 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#350 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#520 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #350, #340);
#530 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#660, #350, #340);

/* type discriminator for part as product */
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#380, #680, #440,
#640));

/* part master for component definitions */
#360 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#380, #680));
#380 = PRODUCT('g1', 'gasket', $, (#220));
#390 = PRODUCT_DEFINITION_FORMATION('A','',#380);
#400 = PRODUCT_DEFINITION('gv1', 'design view on gasket', #390, #230);

#680 = PRODUCT('c1', 'cylinder', $, (#220));
#690 = PRODUCT_DEFINITION_FORMATION('A','',#680);
```

```
#700 = PRODUCT_DEFINITION('cv1', 'design view on cylinder', #690,
#230);

/* part master for sub-assemly definition */
#420 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $, (#440, #640));
#440 = PRODUCT('s1', 'sleeve assembly', $, (#220));
#450 = PRODUCT_DEFINITION_FORMATION('A','',#440);
#460 = PRODUCT_DEFINITION('sv1', 'design view on sleeve assembly',
#450, #230);

/* part master for higher level hub assemly definition */
#640 = PRODUCT('h1', 'hub assembly', $, (#220));
#650 = PRODUCT_DEFINITION_FORMATION('A','',#640);
#660 = PRODUCT_DEFINITION('hv1', 'design view on hub assembly', #650,
#230);

/* single occurrence of component (cylinder) in subassembly defn */
#520 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('cu-1', 'single instance usage',
   'cu1', #460, #700, $);

/* multiple single occurrences of component (gasket) in subassembly */
/* the left gasket in a sleeve subassembly */
#530 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu-1', 'single instance usage',
   'gu1 left', #460, #400, $);
/* the right gasket in a sleeve subassembly */
#540 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu-2', 'single instance usage',
   'gu2 right', #460, #400, $);

/* multiple single occurrences of subassembly defn in assembly defn */
/* the rear sleeve subassembly in a hub assembly */
#550 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('su-1', 'single instance usage',
   'su1 rear', #660, #460, $);
/* the front sleeve subassembly in a hub assembly */
#560 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('su-2', 'single instance usage',
   'su2 front', #660, #460, $);

/* specified usage of individual components in higher level assembly */
/* the right gasket in the front sleeve subassembly */
#570 = SPECIFIED_HIGHER_USAGE_OCCURRENCE('gu-2b', 'single instance
usage', 'gu2b front right', #660, #400, $, #560, #540);
/* the cylinder in the rear sleeve subassembly */
#580 = SPECIFIED_HIGHER_USAGE_OCCURRENCE('cu-1', 'single instance
usage', 'cu1 rear', #660, #700, $, #550, #520);
```

Example 16 : exchange file for multi-level assembly DMU

3.3 Different Views on Assembly Structure

Different view definitions of the same version of a part may participate in different explicit product structures in the PDM Schema. For example, a design/as-planned view of a particular version of a part, representing the design discipline part definition, may be engaged in an explicit design assembly structure. A manufacturing/as-built view of the same part version represents the definitional template for the actual physical part, and may participate in a manufactured assembly structure different from the design structure. Finally, a support/as-maintained view of the part version representing the physical part definition that may participate in yet another different disassembly structure. Figure 8 illustrates two different views on a part structure.

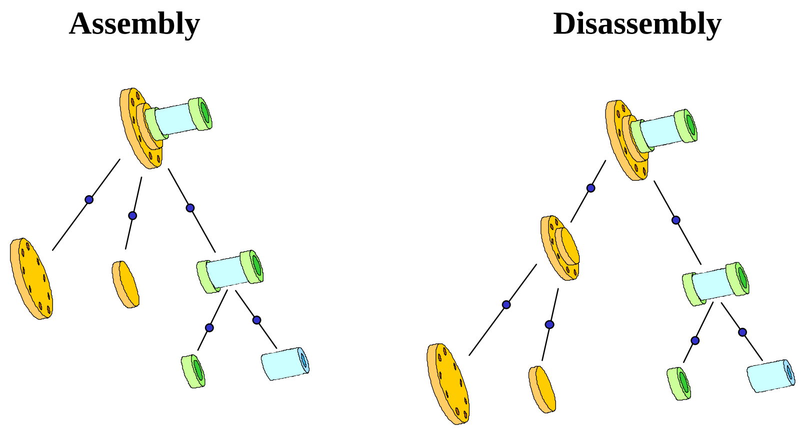
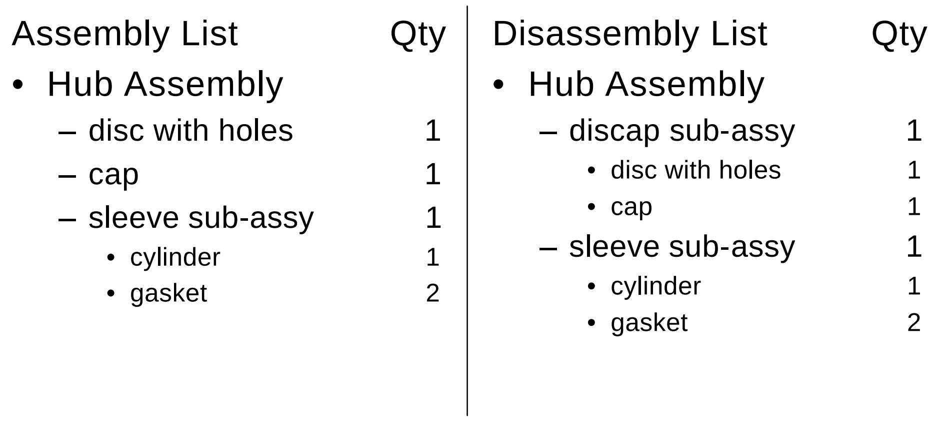


Figure 8 : Different Views on Assembly Structure

Many of the parts that make up the 'nodes' in the two different assembly structures in Figure 8 have the same base identification - they simply have multiple definitions from different life cycle viewpoints. The disc with holes, the cap, and the sleeve sub-assembly are all present in both views of the part structure. However, in this example there is also a new 'node' in the disassembly structure. It corresponds to a sub-assembly, composed of a disc with holes and a cap, which was not present in the assembly structure (see Figure 9).



Assembly List	Qty
• Hub Assembly	
– disc with holes	1
– cap	1
– sleeve sub-assy	1
• cylinder	1
• gasket	2

Disassembly List	Qty
• Hub Assembly	
– discap sub-assy	1
• disc with holes	1
• cap	1
– sleeve sub-assy	1
• cylinder	1
• gasket	2

Figure 9 : Different Indentured Lists Corresponding to Different Structure Views

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and relationships that support the requirements of alternate life cycle viewpoints on product structure are the same as those already described for part structures. Recall that the structural element of the part master 'node' is the product_definition entity, representing a view definition of a part version. The previous description presumed that each part master element in the assembly structure had a consistent life cycle view definition - a single assembly structure from a single view definition context.

In this example of multiple part structures, the part master information corresponding to the 'nodes' disc with holes and cap (and sleeve sub-assy plus it's components) each has two part view definitions. One view definition may represent, for example, the design or manufactured assembly structure; the other a maintenance or disposal disassembly structure. Both of the view definitions are related to the same part version. In this way, the information that is common across the different part structures (the base version identification for all 'nodes' that have definition in both structures) is represented only once. Information that is different between the different views, such as associated management data, property information, or in this case product structure itself is managed independently by using different product_definitions.

The assembly structure would be constructed between the product_definition instances that represent the part view definitions within the same appropriate view context, say for example design.

The disassembly structure would also be constructed between product_definition instances representing part view definitions within one appropriate lifecycle stage - but a different one, for example maintenance. Present only in the disassembly structure, the new or "phantom" part is a sub-assembly composed of a disc with holes and a cap - it is not present in the assembly structure. The part master information corresponding to this discap sub-assy need only have a single part view definition in the appropriate lifecycle stage, e.g. maintenance. The complete disassembly structure is constructed using this product_definition, along with those of the other component part masters for that same view context.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context for design assembly life cycle */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');

/* primary application context for maintenance disassembly view */
#110 = APPLICATION_CONTEXT('maintenance planning');
#240 = PRODUCT_DEFINITION_CONTEXT('part definition', #110,
'maintenance');

/* optional AP214 specific characterization for assembly definitions */
#340 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#350 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#820 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #350, #340);
#830 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#660, #350, #340);

/* type discriminator for part as product */
#100 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', $, (#380, #680, #440,
#640));
#360 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#380, #680));
#420 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $, (#440, #640));

/* part master for common components, each with two view definitions */
/* part master for disc with holes - two view definitions */
#380 = PRODUCT('d1', 'disc with holes', $, (#220));
#390 = PRODUCT_DEFINITION_FORMATION('A','',#380);
#400 = PRODUCT_DEFINITION('dvd1', 'design view on disc', #390, #230);
#410 = PRODUCT_DEFINITION('mvd1', 'maintenance view on disc', #390,
#240);

/* part master for cap component - two view definitions */
#680 = PRODUCT('cap1', 'cap for hub', $, (#220));
#690 = PRODUCT_DEFINITION_FORMATION('A','',#680);
#700 = PRODUCT_DEFINITION('dvc1', 'design view on cap', #690, #230);
#710 = PRODUCT_DEFINITION('mvc1', 'maintenance view on cap', #690,
#240);

/* part master for "phantom" sub-assemly definition - one view only */
#440 = PRODUCT('p1', 'hubcap subassembly', $, (#220));
#450 = PRODUCT_DEFINITION_FORMATION('A','',#440);
#460 = PRODUCT_DEFINITION('mvp1', 'maintenance view hubcap
disassembly', #450, #240);

/* part master for higher level hub assembly - two view definitions */
#640 = PRODUCT('h1', 'hub assembly', $, (#220));
#650 = PRODUCT_DEFINITION_FORMATION('A','',#640);
#660 = PRODUCT_DEFINITION('dvh1', 'design view on hub assembly', #650,
#230);
#670 = PRODUCT_DEFINITION('mvh1', 'maintenance view on hub assembly',
#650, #240);

/* design life cycle 'as planned' assembly structure */
/* single occurrence of component (disc) defn in hub assembly defn */
#520 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('d-1', 'single instance usage',
   'du-1', #660, #400, $);
/* single occurrence of component (cap) defn in hub assembly defn */
#530 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('c-1', 'single instance usage',
   'cu-1', #660, #700, $);

/* maintenance life cycle planned disassembly structure */
/* single "phantom" hubcap subassembly occurrence in hub disassembly */
#560 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('hc-1', 'single instance usage',
   'hc-1', #670, #460, $);
/* single occurrence of component (disc) defn in hubcap sub-assembly */
#540 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('d-1', 'single instance usage',
   'du-1', #460, #410, $);
/* single occurrence of component (cap) defn in hubcap sub-assembly */
#550 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('c-1', 'single instance usage',
   'cu-1', #460, #710, $);
```

Example 17: exchange file for different assembly and disassembly structures

3.4 Other Relationships Between Parts

3.4.1 Alternate Parts

STEP designates alternate and substitute parts differently. Alternate parts are interchangeable in all occurrences whereas substitutes are interchangeable only in a particular usage. Alternate parts in STEP are defined through the alternate_product_relationship entity. This relationship is used in the definition of parts list data for alternate item designations.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of alternate part identification are illustrated in Diagram 24.

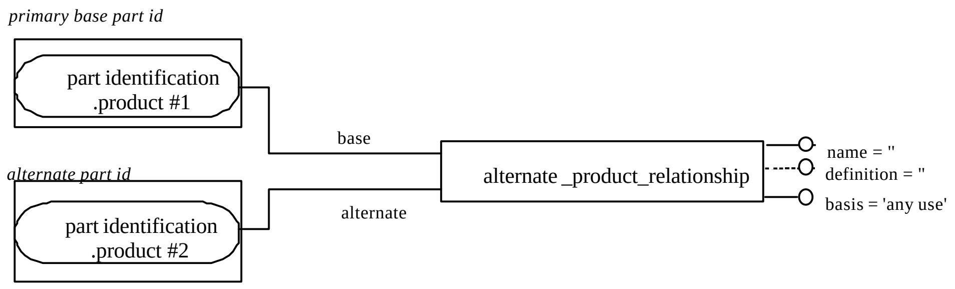

primary base part id
part identification .product #1
alternate part id
part identification .product #2
base
alternate
alternate _product_relationship
name = ''
definition = ''
basis = 'any use'

Diagram 24 : Alternate Part Instance Diagram

3.4.1.1 alternate_product_relationship

This entity represents the alternate part relationship. The alternate part is interchangeable with the base part in any/all uses - it is a context independent alternate that is form, fit, and function equivalent in any use.

Attributes

- The ***name*** attribute has no standard mapping in the PDM Schema.
- The ***definition*** attribute is an optional description.

- The ***base*** attribute is a reference to the primary product of which an alternate is being identified.
- The ***alternate*** attribute is a reference to the product identified as an alternate for the base product.
- The ***basis*** attribute is a rationale for the part interchange (e.g. any use, first available, etc.).

ENTITY alternate_product_relationship	Attribute Population	Remarks
name	type : label = string	
definition	type : text = string	OPTIONAL
base	type : entity = product	
alternate	type : entity = product	
basis	type : text = string	

Pre-processor Recommendations: There are no standard mappings for the name and definition attributes of alternate_product_relationship. The basis attribute should contain a rationale for the part interchange (e.g. any use, first available, etc.) if it is known.

Post-processor Recommendations: Since there are no standard mappings for the name and definition attributes of alternate_product_relationship, it is recommended that post-processors not assign any general processing significance to these values.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context for design assembly life cycle */
#10 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #10, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, 'design');

/* part type discrimination */
#6=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#4, #11));
#7=PRODUCT_CATEGORY_RELATIONSHIP('part-to-detail',$,#6,#8);
#8=PRODUCT_RELATED_PRODUCT_CATEGORY('detail',$,(#4, #11));

#4=PRODUCT('11111','Solid Box','description for part 11111',(#220));
#11=PRODUCT('11011','Solid Cube','description for part 11011',(#12));

#14=ALTERNATE_PRODUCT_RELATIONSHIP('',$,#11,#4,'interchangeable as
solid squares');
```

Example 18 : exchange file segment for alternate part

3.4.2 Substitute Components in an Assembly

Substitute components in an assembly are identified as a relationship between the occurrences of two part definitions (the two substitutes) that are used within the same parent assembly definition. As opposed to the alternate part relationship which is context free, this substitute relationship is context dependent - the substitute is only valid in the context of it's usage as a component within the specified parent assembly definition.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of substitute part identification are illustrated in Diagram 25.

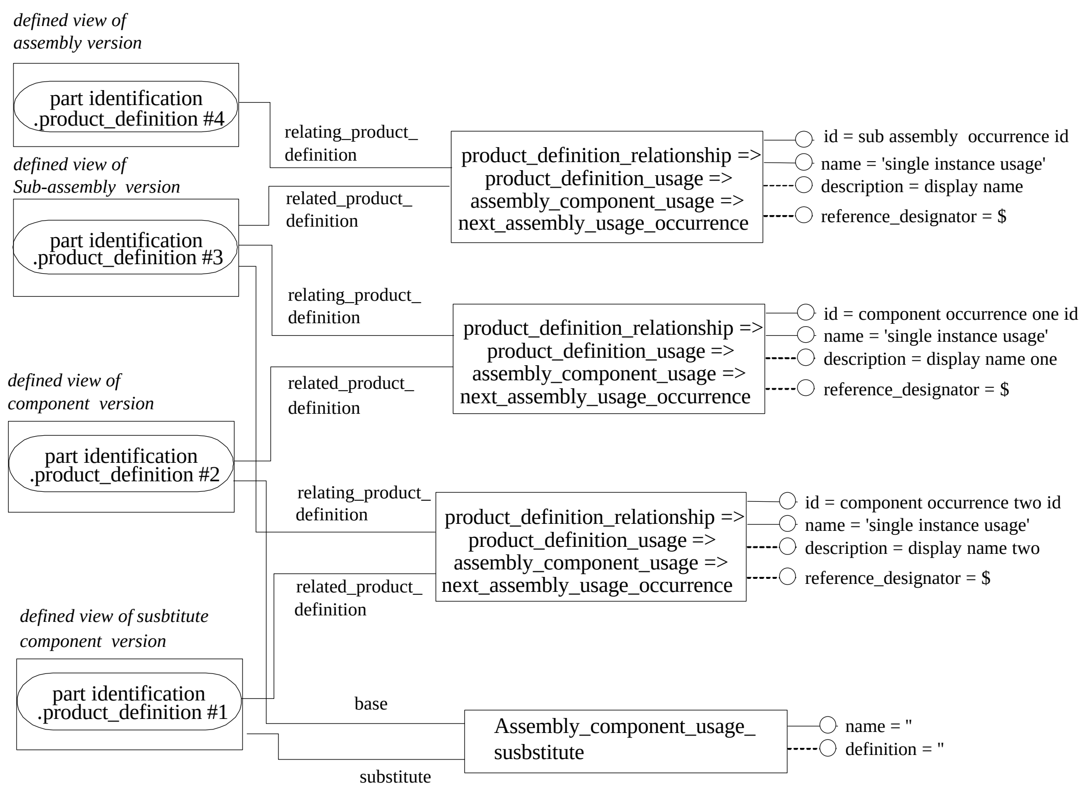

defined view of assembly version
part identification .product_definition #4
defined view of Sub-assembly version
part identification .product_definition #3
defined view of component version
part identification .product_definition #2
defined view of susbtitute component version
part identification .product_definition #1
relating_product_definition
related_product_definition
product_definition_relationship => product_definition_usage => assembly_component_usage => next_assembly_usage_occurrence
id = sub assembly occurrence id
name = 'single instance usage'
description = display name
reference_designator = $
id = component occurrence one id
name = 'single instance usage'
description = display name one
reference_designator = $
id = component occurrence two id
name = 'single instance usage'
description = display name two
reference_designator = $
base
substitute
Assembly_component_usage_susbstitute
name = ''
definition = ''

Diagram 25: Substitute Component Instance Diagram

3.4.2.1 Assembly_component_usage_substitute

This entity represents the substitute component relationship. The substitute part is interchangeable with the base part only in the particular, specified usage - it is a context dependent substitution that is allowed only in the specific usage as the component in the identified component-assembly relationship.

Attributes

- The ***name*** attribute has no standard mapping in the PDM Schema.
- The ***definition*** attribute is an optional description.
- The ***base*** attribute is a reference to the primary component occurrence of which a substitute is being identified.
- The ***substitute*** attribute is a reference to the product identified as a substitute for the base component.

ENTITY assembly_component_usage_substitute	Attribute Population	Remarks
name	type : label = string	
definition	type : text = string	OPTIONAL
base	type : entity = assembly_component_usage	
substitute	type : entity = assembly_component_usage	

Pre-processor Recommendations: There are no standard mappings for the name and definition attributes of assembly_component_usage_substitute. The relating_product_definition for both the base and substitute attributes of this relationship must be the same instance representing the same assembly definition

Post-processor Recommendations: Since there are no standard mappings for the name and definition attributes of assembly_component_usage_substitute, it is recommended that post-processors not assign any general processing significance to these values.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#1330 = PROMISSORY_USAGE_OCCURRENCE('seal_in_sleeve',
   'single instance usage',
   'promissory usage of sealing for sleeve assembly ver2', #1210,
#1300,
   $);
#1335 = PROMISSORY_USAGE_OCCURRENCE('seal_in_sleeve',
   'single instance usage',
   'promissory usage of sealing ver 2 for sleeve assembly ver2', #1210,
#1305,
   $);
#1337 =
ASSEMBLY_COMPONENT_USAGE_SUBSTITUTE('sealings_alternative_sleeve',
   $,#1330,#1335);
```

Example 19 : exchange file segment for substitute component

3.4.3 Make From Relationships

In STEP, the fact that a part is manufactured from another part is indicated by make_from_usage_option. The make_from_usage_option relates the source part definition in the related_product_definition attribute to the resultant part definition in the relating_product_definition attribute.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of make from part identification are illustrated in Diagram 26.

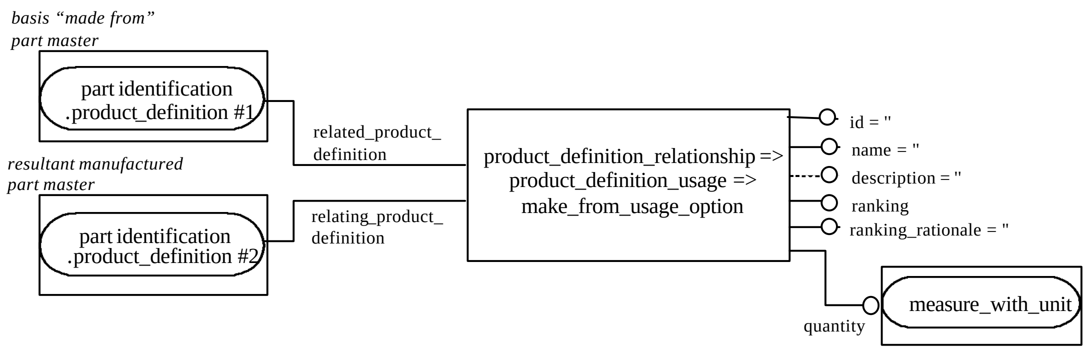

basis "made from" part master
part identification .product_definition #1
resultant manufactured part master
part identification .product_definition #2
related_product_ definition
relating_product_ definition
product_definition_relationship => product_definition_usage => make_from_usage_option
id = ''
name = ''
description = ''
ranking
ranking_rationale = ''
quantity
measure_with_unit

Diagram 26 : Make From Relationship Instance Diagram

3.4.3.1 make_from_usage_option

This entity is a subtype of product_definition_usage. The related_product_definition represents the raw or semi-finished material, and the relating_product_definition represents the resultant manufactured item.

Attributes

- The *relating_product_definition* represents the part manufactured from the related_product_definition.
- The *related_product_definition* represents the raw or semi-finished material.
- The *ranking* attribute is a relative ranking value. A lower value indicates a higher preference.
- The *ranking_rationale* explains the reason for the ranking.
- The *quantity* attribute records the number of resultant parts that can be made from the source part.

ENTITY make_from_usage_option	Attribute Population	Remarks
id	type: identifier = string	unique w.r.t. the relationship
name	type: label = string	
description	type: text = string	OPTIONAL
relating_product_definition	type : entity = product_definition	
related_product_definition	type : entity = product_definition	
ranking	type : integer	
ranking_rationale	type: text = string	
quantity	type: entity = measure_with_unit	

Pre-processor Recommendations: The id attribute must be unique among product_definition_relationship. There is no standard mapping for the name or ranking_rationale attributes in a make_from_usage_option.

Post-processor Recommendations: Since there is no standard value for the name and ranking_rationale attributes for a make_from_usage_option, it is recommended that post-processors not assign any processing significance to these values.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context for design assembly life cycle */
#1000 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #1000, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #1000, 'design');

/* part type discrimination */
#100=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#2, #9));

#2=PRODUCT('11000','solid cube','description for part 11000',(#220));
#3=PRODUCT_DEFINITION_FORMATION('A', 'description of version A for
solid cube',#2);
#4=PRODUCT_DEFINITION('D1','detailed drawing as planned for STEP
conformance testing',#3,#230);

#9=PRODUCT('11111','Solid Box','part 11111 made from raw material
11000',(#220));
#10=PRODUCT_DEFINITION_FORMATION('A', 'description of version A for
part 11111',#9);
#11=PRODUCT_DEFINITION('D2','detailed drawing',#10,#230);
```

```
#16=MAKE_FROM_USAGE_OPTION('id','name','make solid box from two solid
cubes', #11, #4,1,'',#17);
#17 = MEASURE_WITH_UNIT(COUNT_MEASURE(2), #18);
#18 = NAMED_UNIT(#19);
#19 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
```

Example 20 : exchange file for make from relationship

3.4.4 Supplied Part Identification

Supplied part identification is realized by an 'alias relationship' concept. It is specific for supplied parts that allows the renumbering of vendor parts.

In all realms of design and manufacturing business, it is common to buy parts from a vendor and renumber them under an internal numbering scheme. In today's practice, this is done through envelope, specification and source control drawings.

- An envelope drawing is used for a simple renumber of a part where the part is referenced on the envelope drawing and assigned a new part number via the associated parts list.
- A specification control drawing renumbers a part to show that it meets or exceeds the specifications defined on the drawing and to recommend sources for the part.
- A source control drawing renumbers a part and creates a restricted list of suppliers that are qualified to produce the part based on the specifications.

All of these relationships are supported by the product_definition_formation_relationship entity. This entity is used for the identification of part suppliers. The product_definition_formation_relationship may be used for the renumbering of parts. The supplied part relationship relates the "new" part master product_definition_formation in the relating_product_definition_formation attribute to the "old" part master product_definition_formation in the related_product_definition_formation attribute.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements of supplied part identification are illustrated in Diagram 27.

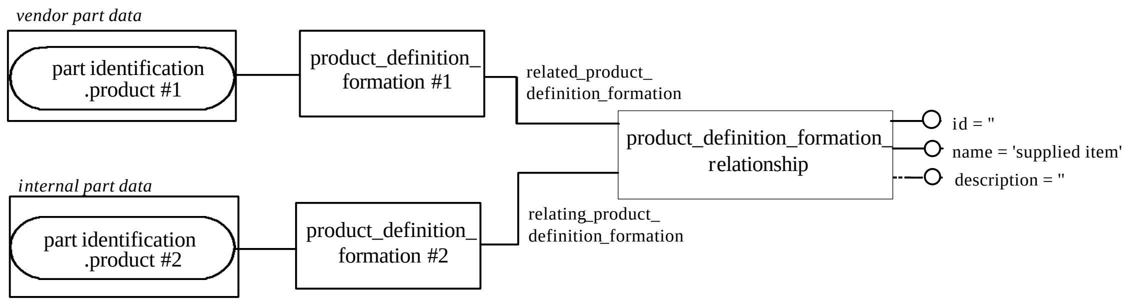

vendor part data
part identification .product #1
product_definition_ formation #1
related_product_ definition_formation
product_definition_formation_ relationship
id = ''
name = 'supplied item'
description = ''
internal part data
part identification .product #2
product_definition_ formation #2
relating_product_ definition_formation

Diagram 27 : Supplied Part Instance Diagram

3.4.4.1 product_definition_formation_relationship

This entity generally relates two product_definition_formation instances in a specified relationship type - the identification of the relationship type is given in the name attribute.

Attributes

- The ***id*** attribute provides an identifier of the relationship.
- The ***name*** attribute specifies this relationship represents 'supplied item'.
- The ***relating_product_definition_formation*** attribute references the internal part version information.
- The ***relating_product_definition_formation*** attribute references the supplied part version information.

ENTITY product_definition_formation_ relationship	Attribute Population	Remarks
id	type: identifier = string	
name	type: label = string	shall be 'supplied item'
description	type: text = string	OPTIONAL
related_product_definition_formation	type : entity = product_definition_formation	supplied part version information
relating_product_definition_formation	type : entity = product_definition_formation	internal part version

Pre-processor Recommendations: The value 'supplied item' for the name attribute identifies this as the supplied item relationship. There is no standard mapping for the description attribute in a product_definition_formation_relationship.. The id attribute must be unique with respect to the relationship, but there is no standard mapping for the value.

Post-processor Recommendations: Since there are no standard mapping for the description attributes for a product_definition_formation_relationship, it is recommended that post-processors not assign any processing significance to this value.

Related Entities: Certification of suppliers can be indicated through the supplied item relationship. This is done by relating an applied_certification_assignment to the product_definition_formation_relationship which associates a certification to the relationship. The applied_certification_assignment entity may have a role associated with it through the entity role_association and its related object_role entity. There are no standard mappings for the values of the role, or the name and purpose attributes of the certification entity.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context for design assembly life cycle */
#1000 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #1000, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #1000, 'design');

/* part type discrimination */
#100=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#2, #9));

/* solid cube - supplied part */
#2=PRODUCT('11000','solid cube','description for part 11000',(#220));
#3=PRODUCT_DEFINITION_FORMATION('A', 'description of version A for
solid cube',#2);
#4=PRODUCT_DEFINITION('D1','detailed drawing as planned for STEP
conformance testing',#3,#230);

/* solid box - internal part */
#9=PRODUCT('11111','Solid Box','part 11111 made from raw material
11000',(#220));
#10=PRODUCT_DEFINITION_FORMATION('A', 'description of version A for
part 11111',#9);
```

```
#11=PRODUCT_DEFINITION('D2','detailed drawing',#10,#230);

#16=PRODUCT_DEFINITION_FORMATION_RELATIONSHIP('id','supplied
item',$,#10,#3);
```

Example 21 : exchange file for supplied part

3.4.5 Version History Relationships

There are two types of relationships between product versions used to represent the version history:

- Sequential relationship: The relating part version is the preceding version and the related part version is the subsequent following version. Both versions must be associated with the same part master base. A part version may have at most one preceding and one subsequent following version.
- Hierarchical relationship: The related part version is a sub version of the relating part version (sometimes referred to as an *iteration*). Both versions must be associated with the same part master base. Each part version may have at most one preceding version. A part version may have an arbitrary number of subsequent following versions.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of part version sequence history are illustrated in Diagram 28.

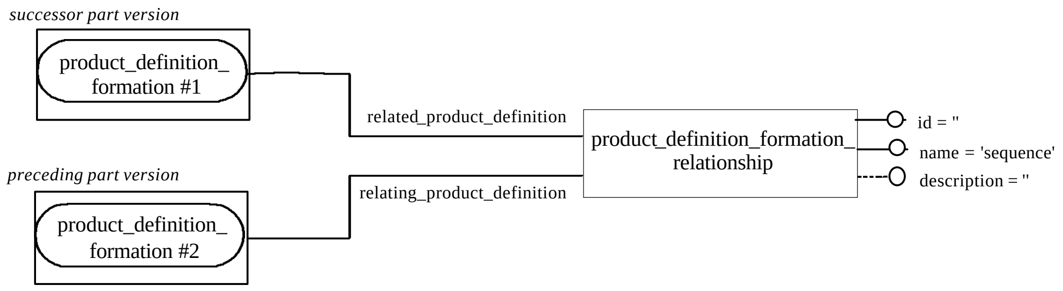

successor part version
product_definition_formation #1
preceding part version
product_definition_formation #2
related_product_definition
relating_product_definition
product_definition_formation_relationship
id = ''
name = 'sequence'
description = ''

Diagram 28 : Part Version (sequence) History Instance Diagram

3.4.5.1 product_definition_formation_relationship

Entity generally relates two product_definition_formation instances in a specified relationship type - the identification of the relationship type is given in the name attribute.

Attributes

- The ***id*** attribute provides an identifier of the relationship.
- The ***name*** attribute specifies this relationship represents a version 'sequence' history.
- The ***relating_product_definition_formation*** attribute references the preceding part version .
- The ***relating_product_definition_formation*** attribute references the subsequent part version.

ENTITY product_definition_formation_relationship	Attribute Population	Remarks
id	type: identifier = string	
name	type: label = string	shall be 'sequence'

description	type: text = string	OPTIONAL
related_product_definition_formation	type : entity = product_definition_formation	subsequent part version
relating_product_definition_formation	type : entity = product_definition_formation	preceding part version

Pre-processor Recommendations: The value 'sequence' for the name attribute identifies this as the version sequence history relationship. There is no standard mapping for the description attribute in a product_definition_formation_relationship.. The id attribute must be unique with respect to the relationship, but there is no standard mapping for the value.

Post-processor Recommendations: Since there are no standard mapping for the description attributes for a product_definition_formation_relationship, it is recommended that post-processors not assign any processing significance to this value.

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* primary application context for design assembly life cycle */
#1000 = APPLICATION_CONTEXT('mechanical design');
#220 = PRODUCT_CONTEXT('', #1000, '');
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #1000, 'design');

/* part type discrimination */
#100=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#2));

/* version A - preceding version */
#2=PRODUCT('11000','solid cube','description for part 11000',(#220));
#3=PRODUCT_DEFINITION_FORMATION('A', 'version A for solid cube',#2);
#4=PRODUCT_DEFINITION('D1','detailed drawing as planned for STEP
conformance testing',#3,#230);

/* version B - subsequent version */
#10=PRODUCT_DEFINITION_FORMATION('B', 'version B for part 11000',#2);
#11=PRODUCT_DEFINITION('D2','detailed drawing',#10,#230);

#16=PRODUCT_DEFINITION_FORMATION_RELATIONSHIP('id','sequence',$,#3,#10)
;
```

Example 22 : exchange file for part version (sequence) history

3.5 Complete instantiation example for part structure and relationships

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('Assembly instantiation example'), '2;1');
FILE_NAME('', '11.06.1999, 08:29:45', (''), (''), '', '', '');
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
DATA;

/* Entities #10 to #520 define that items with associated meta-   */
/* information that are referenced in the product structure       */
/* definitions below                                              */
```

```
#10 = APPLICATION_CONTEXT('');
#20 = PRODUCT_CONTEXT('', #10, '');
#30 = PRODUCT('dca', 'disc cap assembly',
   'disc cap assembly in dissassembly view of hub assembly', (#20));
#40 = PRODUCT('h1', 'hub assembly', '', (#20));
#50 = PRODUCT('s1', 'sleeve assembly', '', (#20));
#60 = PRODUCT_RELATED_PRODUCT_CATEGORY('assembly', $, (#40, #50, #30));
#70 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', '', (#40, #50, #30));
#80 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #70, #60);
#90 = PRODUCT('seal1', 'sealing', '', (#20));
#100 = PRODUCT('d1', 'disc', 'disc with holes', (#20));
#110 = PRODUCT('c1', 'cap', 'cap for hub assembly', (#20));
#120 = PRODUCT('cy1', 'cylinder', 'cyclinder for sleeve assembly',
(#20));
#130 = PRODUCT('g1', 'gasket', 'gasket for sleeve assembly', (#20));
#140 = PRODUCT('gex', 'gasket extern', 'externally supplied gasket', (
   #20));
#150 = PRODUCT('mat23', 'raw material for caps', '', (#20));
#160 = PRODUCT('cl_alt', 'cylinder alternative',
   'alternate product for cy1', (#20));
#170 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', '', (#40, #100, #110,
   #50, #120, #130, #30, #140, #150, #160, #90));
#180 = APPLICATION_PROTOCOL_DEFINITION('version 1.1', 'pdm_schema',
1999
   , #10);
#190 = PRODUCT_RELATED_PRODUCT_CATEGORY('detail', $, (#100, #110, #120,
   #130, #140, #160, #90));
#200 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', '', (#100, #110, #120,
   #130, #140, #160, #90));
#210 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #200, #190);
#220 = PRODUCT_DEFINITION_CONTEXT_ROLE('', $);
#230 = PRODUCT_DEFINITION_CONTEXT('part definition', #10, '');
#240 = PRODUCT_DEFINITION_FORMATION('next_ass',
   'hub assembly version 2 (with individual occurrences of components',
#40);
#250 = PRODUCT_DEFINITION('view 1 hub', 'design view on hub assembly',
   #240, #230);
#260 = APPLICATION_CONTEXT('mechanical design');
#270 = PRODUCT_DEFINITION_CONTEXT('', #260, 'design');
#280 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#250, #270, #220);
#290 = PRODUCT_DEFINITION_FORMATION('1', '', #100);
#300 = PRODUCT_DEFINITION('disc view 1', 'design view on disc', #290,
   #230);
#310 = APPLICATION_CONTEXT('mechnical design');
#320 = PRODUCT_DEFINITION_CONTEXT('', #310, 'design');
#330 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#300, #320, #220);
#340 = PRODUCT_DEFINITION_CONTEXT_ROLE('part definition type', $);
#350 = PRODUCT_DEFINITION_CONTEXT('assembly definition', #10, '');
#360 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#250, #350, #340);
#370 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('disc1', 'single instance usage',
'', #250, #300, $);
#390 = PRODUCT_DEFINITION_FORMATION('1', '', #130);
#400 = PRODUCT_DEFINITION('gv1', 'design view on gasket', #390, #230);
#410 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#400, , #220);
#420 = PRODUCT_DEFINITION_FORMATION('1', '', #120);
#430 = PRODUCT_DEFINITION('cv1', 'design view on cylinder', #420,
#230);
```

```
#440 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#430, #270, #220);
#450 = PRODUCT_DEFINITION_FORMATION('1', '', #50);
#460 = PRODUCT_DEFINITION('sv1', 'design view on sleeve assembly', #450
   , #230);
#470 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #270, #220);
#480 = PRODUCT_DEFINITION_FORMATION('1', '', #110);
#490 = PRODUCT_DEFINITION('capv1', 'design view on cap', #480, #230);
#500 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#490, #270, #220);
#520 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#460, #350, #340);


/* Instances #530 to #570 define the product structure for        */
/* the hub assembly of #250 via the comepoents including the      */
/* sub-assembly of the sleeve #460                                */
#530 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu1', 'single instance usage',
   'gasket usage 1', #460, #400, $);
#540 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu2', 'single instance usage',
   'gasket usage 2', #460, #400, $);
#550 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('cyl1', 'single instance usage',
   'cyclinder usage', #460, #430, $);
#560 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('cap1', 'single instance usage',
   'usage of cap in hub assembly', #250, #490, $);
#570 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('svu1', 'single instance usage',
   'usage of sleeve sub-assembly in hub assembly', #250, #460, $);


/* Entites #580 to #700 define the 'id owners' for the parts  */
#580 = ORGANIZATION('com1', 'Xample Company', 'company');
#590 = ORGANIZATION_ROLE('id_owner');
#600 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #590, (#40));
#610 = ORGANIZATION_ROLE('id owner');
#620 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #610, (#130));
#630 = ORGANIZATION_ROLE('id owner');
#640 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #630, (#120));
#650 = ORGANIZATION_ROLE('id owner');
#660 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #650, (#50));
#670 = ORGANIZATION_ROLE('id owner');
#680 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #670, (#110));
#690 = ORGANIZATION_ROLE('id owner');
#700 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #690, (#100));


/* Entity #710 is an alternate view on the hub assembly version   */
/* defined by the product_definition_formation #240               */
/* The product structure in this view is defined by the entities  */
/* #760 to #810                                                   */
/* This product structure is different to the view #240 of the    */
/* same part version defined by #250. In this example the alternate */
/* product structure related to the view is a dissassembly structure */
/* as opposed to the assembly structure related to the view #250   */
#710 = PRODUCT_DEFINITION('view 2 hub', 'dissassembly view on hub',
#240, #230);
#720 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#710, $, #220);
#730 = PRODUCT_DEFINITION_FORMATION('1', '', #30);
#740 = PRODUCT_DEFINITION('dcav1', 'dissassembly view on disc and cap',
   #730, #230);
#750 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#740, $, #220);
```

```
#760 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('du', 'single instance usage',
   'disc in disc cap assembly', #740, #300, $);
#770 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#740, #350, #340);
#780 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('cu', 'single instance usage',
   'cap in disc-cap assembly', #740, #490, $);
#790 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#710, #350, #340);
#800 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('su2', 'single instance usage',
   'sleeve assembly in dissassembly view of hub', #710, #460, $);
#810 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('dcau1', 'single instance usage',
'disc cap assembly usage in dissassembly view of hub', #710, #740, $);


/* #820 is another version of the hub assembly defined by #40     */
/* This version is specified with a product structure that        */
/* quantifies the occurrences of components in an assembly        */
/* The product structure relationships that defines the qunaitifed */
/* usage of the components in an assembly are instaniated complex  */
/* with NEXT_ASSEMBLY_COMPONENT_USAGE to indicate that a given     */
/* parent node directly uses the referenced component              */
#820 = PRODUCT_DEFINITION_FORMATION('quantif',
   'version specified with qunantified  usages', #40);
#830 = PRODUCT_DEFINITION('qv1',
   'view on quantified version of hub assembly', #820, #230);
#840 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#830, $, #220);
#850 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#830, #350, #340);
#860 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#870 = NAMED_UNIT(#860);
#880 = MEASURE_WITH_UNIT(COUNT_MEASURE(1), #870);
#890 = (ASSEMBLY_COMPONENT_USAGE($) NEXT_ASSEMBLY_USAGE_OCCURRENCE()
   PRODUCT_DEFINITION_RELATIONSHIP('du4', 'quantified instance usage',
   'quantified usage of disc', #830, #300) PRODUCT_DEFINITION_USAGE()
   QUANTIFIED_ASSEMBLY_COMPONENT_USAGE(#880));
#900 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#910 = NAMED_UNIT(#900);
#920 = MEASURE_WITH_UNIT(COUNT_MEASURE(1), #910);
#930 = (ASSEMBLY_COMPONENT_USAGE($) NEXT_ASSEMBLY_USAGE_OCCURRENCE()
   PRODUCT_DEFINITION_RELATIONSHIP('cau4', 'quantified instance usage',
'quanitifed usage if cap', #830, #490) PRODUCT_DEFINITION_USAGE()
   QUANTIFIED_ASSEMBLY_COMPONENT_USAGE(#920));
#940 = PRODUCT_DEFINITION_FORMATION('slq',
   'quantifed version of sleeve assembly', #50);
#950 = PRODUCT_DEFINITION('slqv',
   'design view quantified version sleeve assembly', #940, #230);
#960 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#950, $, #220);
#970 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#950, #350, #340);
#980 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#990 = NAMED_UNIT(#980);
#1000 = MEASURE_WITH_UNIT(COUNT_MEASURE(1), #990);
#1010 = (ASSEMBLY_COMPONENT_USAGE($) NEXT_ASSEMBLY_USAGE_OCCURRENCE()
   PRODUCT_DEFINITION_RELATIONSHIP('cu4', 'quantified instance usage',
   'qunatifed usage of cylinder', #950, #430)
PRODUCT_DEFINITION_USAGE()
    QUANTIFIED_ASSEMBLY_COMPONENT_USAGE(#1000));
#1020 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#1030 = NAMED_UNIT(#1020);
#1040 = MEASURE_WITH_UNIT(COUNT_MEASURE(2), #1030);
#1050 = (ASSEMBLY_COMPONENT_USAGE($) NEXT_ASSEMBLY_USAGE_OCCURRENCE()
   PRODUCT_DEFINITION_RELATIONSHIP('gu4', 'quantified instance usage',
   'quantifed usage of gasket for sleeve', #950, #400)
   PRODUCT_DEFINITION_USAGE()
QUANTIFIED_ASSEMBLY_COMPONENT_USAGE(#1040)
   );

/* #140 represents an externally supplied gasket part              */
/* The relationship #1070 relates the version 1 of this externally */
/* supplied part to the internal part master represented by        */
/* #140 in its version 1 (defined by #390) */
#1060 = PRODUCT_DEFINITION_FORMATION('1', '', #140);
#1070 = PRODUCT_DEFINITION_FORMATION_RELATIONSHIP('', 'supplied item',
   'supplied item version of gasket', #1060, #390);
#1080 = ORGANIZATION('com2', 'Supplier Company', 'company');
#1090 = ORGANIZATION_ROLE('id owner');
#1100 = APPLIED_ORGANIZATION_ASSIGNMENT(#1080, #1090, (#1060));

/* product #140 represents the raw material for the cap product  */
/* (#110), version 1 (#480) in the design view (#490) */
#1120 = PRODUCT_RELATED_PRODUCT_CATEGORY('raw material', '', (#150));
#1140 = PRODUCT_DEFINITION_FORMATION('1', '', #150);
#1150 = PRODUCT_DEFINITION('mat_view', 'design view on mat23', #1140,
   #230);
#1160 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1150, $, #220);
#1170 = ORGANIZATION_ROLE('id owner');
#1180 = APPLIED_ORGANIZATION_ASSIGNMENT(#580, #1170, (#1140));

/* Entities #1181 to #1185 establish the relation that defines that  */
/* the cap (in view of #490) is made from the raw material specified */
/* via #1150 and related entities                                    */
#1181 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#1182 = NAMED_UNIT(#1181);
#1184 = MEASURE_WITH_UNIT(COUNT_MEASURE(1), #1182);
#1185 = MAKE_FROM_USAGE_OPTION('mkfr1','','',#490,#1150,1,'',#1184);


/* #1190 models the successor version of the sleeve assembly    */
/* version 1 define by #450. 'Sequence' indicates that #1190 is */
/* a direct successor to #450                                   */
#1190 = PRODUCT_DEFINITION_FORMATION('2',
   'successor version of version 1', #50);
#1200 = PRODUCT_DEFINITION_FORMATION_RELATIONSHIP('', 'sequence', '',
   #450, #1190);


/* #1210 specifies a view on the sleeve assembly version 2. To this  */
/* view a product structure is attached that extends the product     */
/* structure of version one with an additonal sealing part           */
/* For this sealing no detail is given, a                   */
/* promissory_usage_occurrence is used to model the intent to */
/* include the sealing in the sleeve assembly               */
#1210 = PRODUCT_DEFINITION('sv2', 'design view on sleeve version 2',
   #1190, #230);
#1220 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1210, $, #220);
#1230 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1210, #350, #340);
#1240 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu5', 'single instance usage',
   'gasket usage for ver2 of sleeve assembly', #1210, #400, $);
#1250 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('gu6', 'single instance usage',
   '2ns gasket usage for sleeve assembly ver2', #1210, #400, $);
#1260 = PRODUCT_DEFINITION_FORMATION('1', '', #160);
#1270 = PRODUCT_DEFINITION('v1_cyl_alt',
   'design view on alternate cylinder', #1260, #230);
#1280 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1270, $, #220);
#1290 = PRODUCT_DEFINITION_FORMATION('1', 'sealing ver1', #90);
#1295 = PRODUCT_DEFINITION_FORMATION('2', 'sealing ver2', #90);
#1300 = PRODUCT_DEFINITION('sv1', 'design view on sealing v1', #1290,
#230);
#1305 = PRODUCT_DEFINITION('sv2', 'design view on sealing v2', #1295,
#230);
#1310 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1300, $, #220);
#1320 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('cylalt', 'single instance
usage'
   , 'usage of alternate cylcinder ofr sleeve v2', #1210, #1270, $);

/* #1330 models the intent to use the sealing part as defined */
/* by #1300 in the sleeve part as defined by #1210            */
#1330 = PROMISSORY_USAGE_OCCURRENCE('seal_in_sleeve',
   'single instance usage',
   'promissory usage of sealing for sleeve assembly ver2', #1210,
#1300,
   $);


/* The entities #1335 and #1337 define a usage of another version of */
/* the sealing part given by its view in #1305 in the sleeve         */
/* assembly #1210. #1337 specifies that the usage of the two version */
/* of the sealing in the sleeve assembly is alternative, i.e. the    */
/* product structure relationship defined by #1335 is alternative to */
/* the one defined by #1330                                          */
#1335 = PROMISSORY_USAGE_OCCURRENCE('seal_in_sleeve',
   'single instance usage',
   'promissory usage of sealing ver 2 for sleeve assembly ver2', #1210,
#1305,
   $);
#1337 =
ASSEMBLY_COMPONENT_USAGE_SUBSTITUTE('sealings_alternative_sleeve',
   $,#1330,#1335);


/* #1340 is a version of the hub assembly #40 that is derived from */
/* the version #240 of the same part. The derivation relationship is */
/* established by entity #1350                                   */
#1340 = PRODUCT_DEFINITION_FORMATION('ver_double',
   'double sleeve version', #40);
#1350 = PRODUCT_DEFINITION_FORMATION_RELATIONSHIP('', 'derivation', '',
   #240, #1340);
#1360 = PRODUCT_DEFINITION('dsl_view',
   'design view on double sleeve version of hub', #1340, #230);
#1370 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1360, $, #220);
```

```
#1380 = PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#1360, #350, #340);

/* The assembly structure below assembles the #1340 version of the */
/* hub assembly with 2 sleeves. the sleeves each consist of two */
/* gaskets and and one cylinder. Thus in the resulting product */
/* structure there are four cylinders. The basic next_assembly_usage */
/* construct allows only to distinguish between two gaskets one the */
/* level of the sleeve assembly definition. To indivdiulize one of */
/* the gaskets in the hub assembly specified_higer_usage_occurrence */
/* is used.                   */
#1390 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('dcau', 'single instance usage',
'usage of dca assembly for double sleeve version', #1360, #740, $);
#1400 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('usl1', 'single instance usage',
'1st sleeve in double sleeve version', #1360, #460, $);
#1410 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('usl2', 'single instance usage',
'2nd sleeve in double sleeve assembly', #1360, #460, $);

/* #1415 indvidualizes the first gasket (#530) used in the          */
/*  second sleeve (#1410) of the assembly #1360.                    */
#1415 = SPECIFIED_HIGHER_USAGE_OCCURRENCE('2nd_g',
  'identification of second gasket of first sleeve',
  $,
  #1360,    /* the hub assembly  */
  #400,     /* gasket used for the assembly */
  $,        /* reference_designator       */
  #1410,    /* upper_usage: select the second sleeve usage */
  #530      /* next_usage_: select first usage of gasket in sleeve */
  );

/* #1420 specifies that the alternate cylinder part #160 is an     */
/* alternative to the cylinder #120. These parts are mutually      */
/* exchangeable in any context                                     */
#1420 =
ALTERNATE_PRODUCT_RELATIONSHIP('hub_alternativity',$,#160,#120,'text
for basis');
ENDSEC;
END-ISO-10303-21;
```

Example 23 : exchange file for complete part structure and relationships

3.6 Known Issues

3.6.1 Version Hierarchy Relationship

Version history information has been described as the sequence relationship between part versions (see 3.4.5). The entity product_definition_formation_relationship with attribute name value 'sequence' represents this information. In addition, AP214 defines the value 'hierarchy' for this name attribute, to identify a hierarchical relationship between two versions of different levels, e.g. a version and an iteration. This representation of iterations as well as released versions (revisions) of a part is not yet completely understood in relation to the sequence history or the implementation of current PDM systems.

4 Document Identification

The PDM Schema deals with documents as products, according to a basic STEP interpretation of 'Document as Product'. As with 'Part as Product', there are three basic concepts central to document identification in the PDM Schema:

- product master identification,
- context information,
- type classification.

These fundamental concepts are described for 'Part as Product' in this document. The following provides specifics on the distinction and additions for the 'Document as Product' approach.

'Document as Product' identifies a managed document object in a PDM system. A managed document is under revision control, and may distinguish various representation definitions of a document version. The document_version represents the minimum identification of a managed document under revision control. A document representation definition may optionally be associated with one or more constituent external files that make it up.

EXAMPLE - a configuration controlled managed paper document like a drawing would generally map to a usage of the entity 'product', with version identification and a (physical) representation/view definition according to the 'document as product' approach.

External files in the PDM Schema represent a simple external reference to a named file. An external file is not managed independently by the system - there is usually no revision control or any representation definitions of external files. Version identification may optionally be associated with an external file, but this is for information only and is not used for managed revision control.

If a file is under configuration control, it should be represented as a constituent of a document definition view/representation. In this case it is actually the managed document that is under direct configuration control, the file is in this way indirectly under configuration control. A change to the file results in a change to the managed document (i.e., a new version) - the changed file would be mapped as a constituent of a view/representation definition of the new document version. A simple external reference alone is not configuration controlled, it is just an external file reference to product data.

Documents may be associated with product data in a specified role, to represent some relationship between a document and other elements of product data. Constraints may also be specified on this association, in order to distinguish an applicable portion of an entire document or file in the association. With 'Document as Product', additional entities are required to relate a managed document to other product data (see Section 7.1). Included among these is the entity document. The document.id should not be used as valid user data - the document entity does not always need to be instantiated using 'Document as Product', it is done only to assign the document to other product data via applied_document_reference.

4.1 Document as Product

The interpretation of 'Document as Product' uses basic product master identification for the fundamental requirements of document identification, versioning, and representation definition. 'Document as Product' is distinguished from 'Part as Product' in each of the three basic elements of product identification:

- product master identification - document identification has specific requirements to assign documents to other product data, and to optionally associate with the constituent external file(s) that make up a specific document representation view definition;
- context information - document identification has different context information than part identification;

- type classification - document identification has a different type classification than part identification.

4.1.1 Document Master Identification

As with 'Part as Product', the concepts of base identification, version identification, and view definition are structurally distinct in 'Document as Product'. The general recommendations given for part master identification apply to the document master identification, except where differences are noted.

Base document identification is always associated with at least one document version. Multiple document versions of a base document identification may be related together to represent document version history.

With 'Document as Product' the product view definition is used to define a view of a particular representation of a document version. A document version does not have to have an associated document representation definition.

The representation view definition of a document version is used for association of document properties, to build document structures, or to associate a document with the set of constituent external files that make it up.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support identification of a document version are illustrated in Diagram 29. Document version identification without an associated document view representation definition represents the minimum requirements for document master identification.

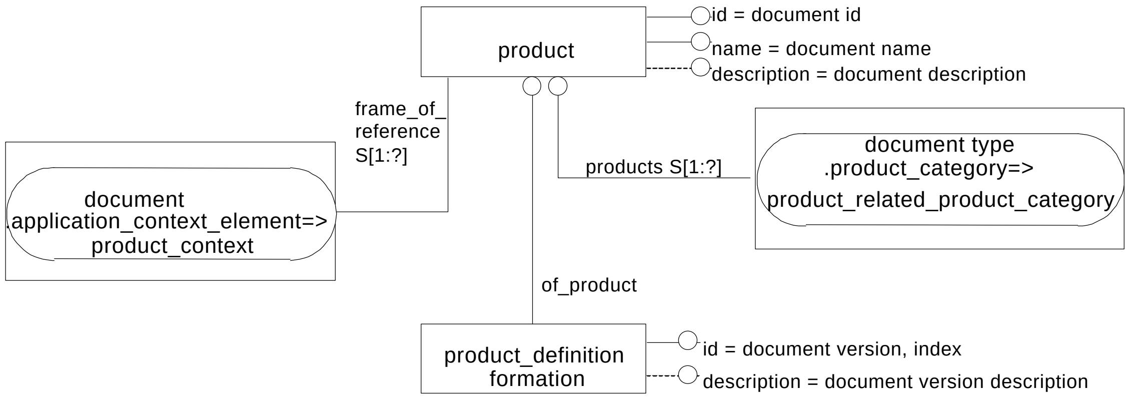

product
id = document id
name = document name
description = document description
frame_of_reference S[1:?]
products S[1:?]
document type .product_category=> product_related_product_category
document .application_context_element=> product_context
of_product
product_definition formation
id = document version, index
description = document version description

Diagram 29: Minimum Document Identification Instance Diagram

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#270=PRODUCT('doc_4711','packaging guide','packaging guideline for
part',(#260));
#280=PRODUCT_DEFINITION_FORMATION('ver3.1','version 3.1',#270);
```

Example 24: exchange file segment for minimum document identification

4.1.2 Context Information

Context information provides a scope and necessary circumstance for product identification information. It consists of two separate and related areas:

- Application Protocol Identification,
- Application Context Information.

Application protocol identification and general context information are handled structurally the same for 'Document as Product' as for 'Part as Product'. A single instance of the entity application_context should be referenced by all product_context entities, for both parts and documents. This application_context should be referenced by the single instance of the entity application_protocol_definition.

The application context information is managed differently to distinguish documents from parts.

- Life-cycle information is not maintained for 'Document as Product', as it is for 'Part as Product' to identify the development life-cycle stages. When a managed document is assigned to a part master, the life-cycle stage of the part master identification may be extrapolated as relevant to the document;
- To distinguish a document representation view definition, the names 'digital document definition' or 'physical document definition' are used, as differentiated from the value 'part definition' used in 'Part as Product'. A digital document definition represents an electronic document, while a physical document definition stands for a 'hardcopy', typically paper, document.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of document identification with context information are illustrated in Diagram 30.

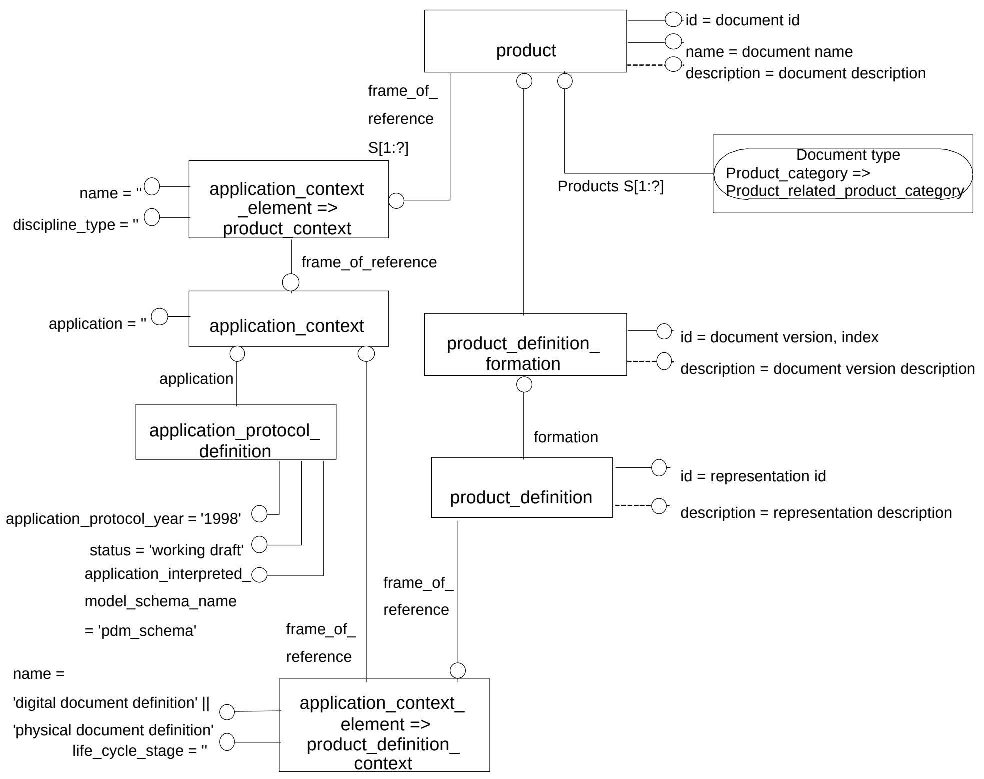

product
id = document id
name = document name
description = document description
frame_of_reference S[1:?]
Products S[1:?]
Document type
Product_category => Product_related_product_category
name = ''
discipline_type = ''
application_context_element => product_context
frame_of_reference
application = ''
application_context
application
product_definition_formation
id = document version, index
description = document version description
application_protocol_definition
formation
product_definition
id = representation id
description = representation description
application_protocol_year = '1998'
status = 'working draft'
application_interpreted_model_schema_name = 'pdm_schema'
frame_of_reference
frame_of_reference
name = 'digital document definition' || 'physical document definition'
life_cycle_stage = ''
application_context_element => product_definition_context

Diagram 30: Document Master with Context Information Instance Diagram

4.1.2.1 product_definition

The product_definition entity denotes the definition of a particular view of a representation of a document version. There may be more than one document representation definition associated with a single document version. The representation view definition of a document version is used for association of document properties, to build document structures, or to associate a document with the set of constituent external files that make it up. The entity product_definition supports property association and document structure. The subtype product_definition_with_associated_documents is used to associate a representation of a document version with the set of constituent files that make it up.

Attributes

- The ***id*** attribute indicates the unique identifier of the document representation definition.
- The ***description*** attribute provides optional additional words describing the representation definition.
- The ***formation*** attribute references the document version of which this is a representation definition.
- The ***frame_of_reference*** attribute references the context information related to this definition.

ENTITY product_definition	Attribute Population	Remarks
id	type : identifier = string	Must be unique in relation with a specific product_version
description	type: text	OPTIONAL
formation	type: entity = product_definition_formation	Document version of which this is a particular document representation definition
frame_of_reference	type : entity = product_definition_context	Context information for this definition.

Preprocessor Recommendations: The value for the id attribute of product_definition should be unique relative to other product_definition entities related to the same product_definition_formation.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

4.1.2.2 product_definition_context

All STEP product_definitions must be defined in a product_definition_context. With 'Document as Product' the context information is used to distinguish digital from physical 'hardcopy' documents.

Attributes

- The ***name*** attribute indicates the type of the document representation definition.
- The ***frame_of_reference*** attribute references application domain information.
- The ***life_cycle_stage*** attribute has no standard mapping for documents in this usage guide.

ENTITY product_definition_context	Attribute Population	Remarks
name	type: label = string 'digital document definition' or 'physical document definition'	Distinguishes the associated product_definition as that of a document
frame_of_reference	type: entity = application_context	
life_cycle_stage	type: label = text	

Preprocessor Recommendations: The name attribute distinguishes the representation definition of a document version as either digital ('digital document definition') or physical, i.e., hardcopy ('physical document definition').

Postprocessor Recommendations: Postprocessors should interpret the value of the name attribute as a distinction between part and document definitions, and further between digital and physical document definitions.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#240=APPLICATION_CONTEXT('');
#245=APPLICATION_PROTOCOL_DEFINITION('version
1.1','pdm_schema',1999,#240);
#250=PRODUCT_DEFINITION_CONTEXT('digital document definition',#240,'');
#260=PRODUCT_CONTEXT('',#240,'');
#270=PRODUCT('doc_4711','packaging guide','packaging guideline for
part',(#260));
#280=PRODUCT_DEFINITION_FORMATION('ver3.1','version 3.1',#270);
#290=PRODUCT_DEFINITION('id of digital document','digital document for
representing guideline ver3.1',#280,#250);
```

Example 25: exchange file segment for document master with context information

4.1.3 Type Classification

Type classification information provides the basic capability to distinguish products interpreted to represent documents from those interpreted as parts.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the complete requirements of document identification with context information and type classification are illustrated in
Diagram 31.

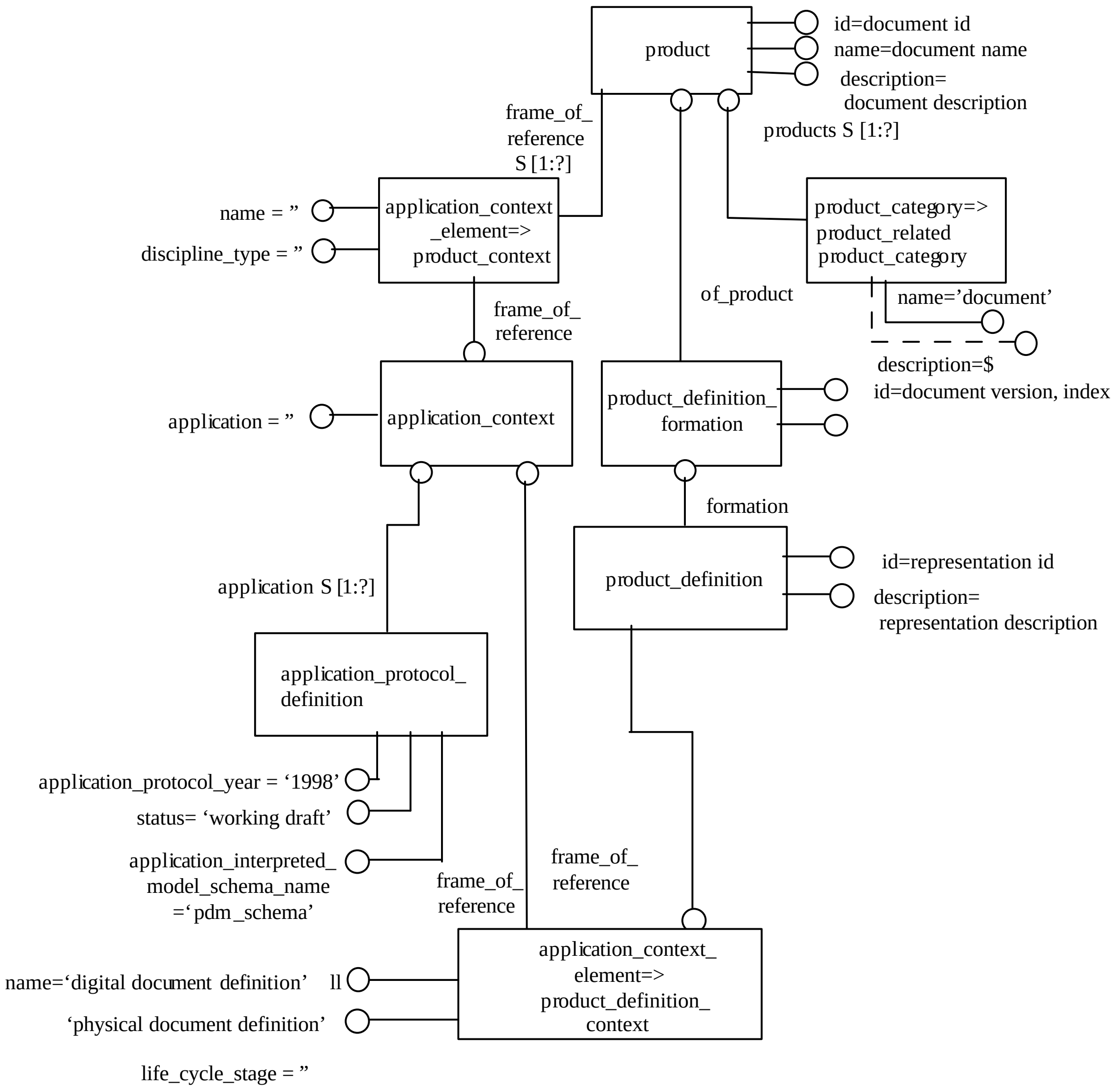

product
id=document id
name=document name
description= document description
products S [1:?]
frame_of_ reference S [1:?]
name = ”
discipline_type = ”
application_context _element=> product_context
product_category=> product_related product_category
name='document'
description=$
of_product
frame_of_ reference
application = ”
application_context
product_definition_ formation
id=document version, index
formation
product_definition
id=representation id
description= representation description
application S [1:?]
application_protocol_ definition
application_protocol_year = '1998'
status= 'working draft'
application_interpreted_ model_schema_name ='pdm_schema'
frame_of_ reference
frame_of_ reference
application_context_ element=> product_definition_ context
name='digital document definition' ll
'physical document definition'
life_cycle_stage = ”

Diagram 31: Complete Document Master with Context and Type Classification Instance Diagram

4.1.3.1 product_related_product_category

The product_related_product_category represents the identification of a specific classification applied to a product. The name and description attributes are inherited from the supertype product_category. This subtype adds the attribute products that allow it to be associated (related) directly to a product instance. In the usage scenario of 'Document as Product', these product instances are used to represent managed documents.

Attributes

- The ***products*** attribute associates the category with the document as product to which it applies.

ENTITY product_related_ product_category	Attribute Population	Remarks
name	type: label = string 'document'	
description	type: text = string	
products	type: entity = product	SET[1:?] of product entity instance(s) distinguished as representing documents.

Preprocessor Recommendations: The name attribute should have the value 'document' to discriminate 'Document as Product' from 'Part as Product'.

Postprocessor Recommendations: When the value of the attribute name is 'document', postprocessors should recognize that the value(s) of the attribute products are the representation of managed documents as opposed to parts.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#240=APPLICATION_CONTEXT('');
#245=APPLICATION_PROTOCOL_DEFINITION('version
1.1','pdm_schema',1999,#240);
#250=PRODUCT_DEFINITION_CONTEXT('digital document definition',#240,'');
#260=PRODUCT_CONTEXT('',#240,'');
#270=PRODUCT('doc_4711','packaging guide','packaging guideline for
part',(#260));
#280=PRODUCT_DEFINITION_FORMATION('ver3.1','version 3.1',#270);
#290=PRODUCT_DEFINITION('id of digital document','digital document for
representing guideline ver3.1',#280,#250);
#300=PRODUCT_RELATED_PRODUCT_CATEGORY('document',$,(#270));
```

Example 26: exchange file for complete document master with context and type classification

5 External Files

External files in the PDM Schema represent a simple external reference to a named file. The external file may identify a digital file or a physical, 'hardcopy' file. As opposed to a managed 'Document as Product', an external file is not managed by the system - there is no capability for managed revision control or any document representation definitions for an external file.

An external file is simply an external reference that may be associated with other product data. Document/file properties may be associated with an external file as with an identified managed document. In the case where properties differ with different versions, the managed 'Document as Product' approach is recommended.

If a file is under configuration control, it should be represented as a constituent of a document definition view/representation according to 'Document as Product'. In this case, is actually the managed document that is under direct configuration control, the file is, in this way, indirectly under configuration control. A change to the file results in a change to the managed document (i.e., a new version) - the changed file would be mapped as a constituent of a view/representation definition of the new document version. A simple external reference alone is not configuration controlled, it is just an external file reference to product data.

While managed revision control representing multiple versions and version history is not available for external files, external files may have an optional version identification providing a string labeling the version of the file.

5.1 External File Identification

Identification of an external file is done using the entity document_file. The document_file entity is a defined subtype of the entity document. It is also a subtype of the entity characterized_object, which allows association of properties to the identification of an external file.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of external file identification are illustrated in Diagram 32 below.

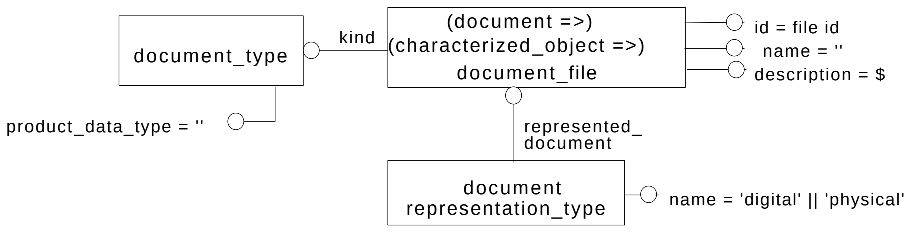


Diagram 32: External File Identification Instance Diagram

5.1.1.1 document_file

This entity is a subtype of the document entity, and thus works together with the applied_document_-reference to support the assignment of external files to product data. Document_file is also a subtype of the entity characterized_object, which has local attributes name and description.

Since both supertypes of this entity define local attributes 'name' and 'description', the subtype document_file has a double inheritance of these attributes. Local constraints defined on the entity document_file specify that the additional attributes inherited from the supertype characterized_object are not to be used for valid user data. These constraints currently require that the value of these attributes be a string with a single space: ' '. This is an identified error in the EXPRESS, it should specify a value of the empty string ''. This EXPRESS error will be corrected in the future. It is recommended that implementations use the empty string ''.

Attributes

- The ***id*** attribute indicates the unique identifier of the external file.
- The ***name*** attribute is double inherited. Only the one derived from the document supertype is valid for user data, the nomenclature of the external file. The name attribute inherited from characterized_object should be assigned the empty string.
- The ***description*** attribute is double inherited. Only the one derived from the supertype document is valid for user data. The description attribute inherited from characterized_object should be assigned the empty string.
- The ***kind*** attribute is a reference to the file "type" classification information.

ENTITY document_file	Attribute Population	Remarks
id	type: identifier = string	
SELF/document.name	type: label = string	valid user data for file id
SELF/document.description	type: text = string	OPTIONAL
kind	type: entity = document_type	
SELF/characterized_object.name	type: label = string empty string ''	not to be used for valid user data
SELF/characterized_object.description	type: text = string empty string ''	not to be used for valid user data

Preprocessor Recommendations: Preprocessors must carefully encode the exchange syntax for an instance of document file to properly handle the multiple inheritance of the attributes name and description. An example instance of document_file is illustrated in Example 27.

The document_file entity requires an associated instance of the entity document_representation_type. The document_file entity constrains the possible values of the attribute document_representation_type.name, as described in 5.1.1.2.

The id attribute is a unique identifier for the external file. Access path information for a file should be represented as a file "source" property.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

5.1.1.2 document_representation_type

This entity provides the capability to identify the type of the representation of a particular external file, either digital or physical.

Attributes

- The ***name*** attribute is used to identify the particular representation type (either digital or physical) of the associated external file.
- The ***represented_document*** attribute is a reference to the associated external file.

ENTITY document_representation_type	Attribute Population	Remarks
name	type: text = string 'digital' or 'physical'	
represented_document	type: entity = document	for file identification, this attribute will reference the subtype document_file

Preprocessor Recommendations: For file_identification, the possible values for the name attribute are 'digital' and 'physical'. A digital file represents an electronic file on a computer system. A physical file is the actual paper hardcopy or other physical realization of a file.

Postprocessor Recommendations: A value for the name attribute of 'digital' should be interpreted to mean the associated document_file represents an external reference to an electronic digital file. The value 'physical' should be interpreted to mean the associated document_file represents an external reference to a hardcopy file.

Related Entities: There are no specific related entities.

5.1.1.3 document_type

This entity provides the capability to identify the type of an external file for the general requirement of file type classification.

Attributes

- The ***product_data_type*** attribute is used to identify the kind of product data in the associated file.

ENTITY document_type	Attribute Population	Remarks
product_data_type	type: identifier = string	

Preprocessor Recommendations: There are no specific preprocessor recommendations.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#510=DOCUMENT_FILE('doc_id','',$,#520,'','');
#520=DOCUMENT_TYPE('unspecified type');
#540=DOCUMENT_REPRESENTATION_TYPE('digital',#510);
```

Example 27: exchange file segment for external file identification

Although managed revision control representing multiple versions and version history is not available for external files, they may have an optional version identification that provides a string labeling the version of the external file. If multiple versions are represented, the 'Document as Product' approach is recommended.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements of external files with optional version identification are illustrated in Diagram 33 below.

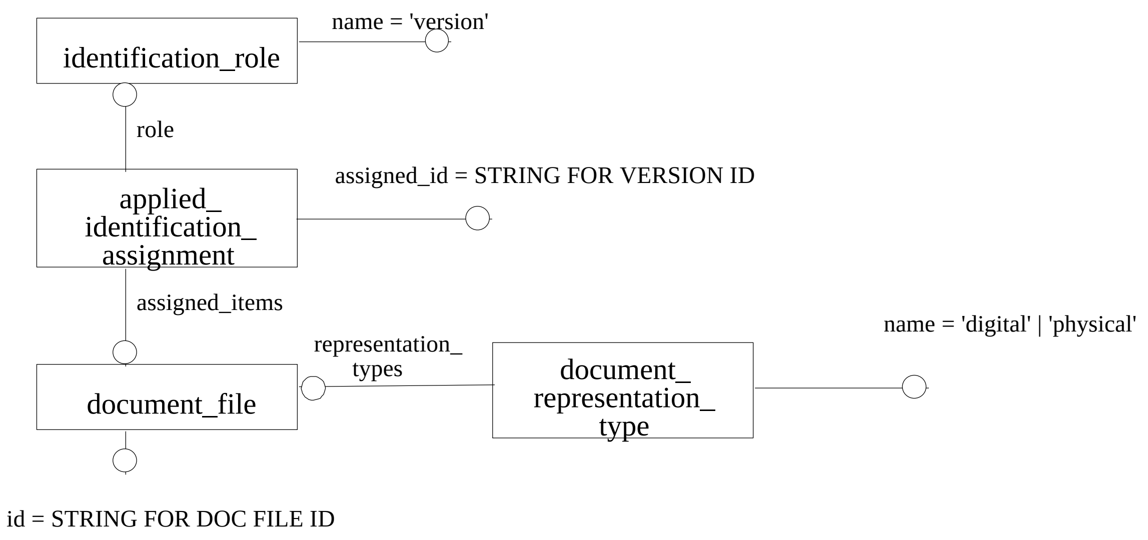

identification_role
name = 'version'
role
applied_ identification_ assignment
assigned_id = STRING FOR VERSION ID
assigned_items
representation_ types
document_file
document_ representation_ type
name = 'digital' | 'physical'
id = STRING FOR DOC FILE ID

Diagram 33: External File with Version Identification Instance Diagram

5.1.1.4 applied_identification_assignment

This entity is a subtype of the entity identification_assignment. It allows the actual assignment of an identification_assignment entity to product data, in this case to a document_file entity representing an external file.

Attributes

- The ***items*** attribute is used to reference the associated external file.

ENTITY applied_identification_assignment	Attribute Population	Remarks
assigned_id	type: identifier = string	contains the string identifying the version of the external file.
role	type: entity = identification_role	for file version identification, this entity should have name attribute value = 'version'
items	type : entity = document_file	reference to the associated external file to which the version identification is assigned.

Preprocessor Recommendations: The use of this entity is optional. When applied to a document_file, it represents simple version identification for the external file. It is not to be used for managed revision control. If managed revision control is the requirement, then the 'Document as Product' approach must be used. The value of the attribute assigned_id contains the version identification. This should not be used for managed version control, but rather for an optional label providing additional information about the version of the document that is identified. The value of the role attribute is an instance of the entity identification_role with it's name attribute assigned the string 'version'.

Postprocessor Recommendations: Postprocessors should recognize this entity as providing version identification label for the associated external file.

Related Entities: There are no specific related entities.

5.1.1.5 identification_role

This entity in this use case indicates the role of a version identification for an external file.

Attributes

- The ***name*** attribute is used to indicate that the related identification_assignment represents a version identification for the associated external file.
- The ***description*** attribute is optional additional text.

ENTITY identification_role	Attribute Population	Remarks
name	type: identifier = string 'version'	Contains the string indicating that the related identification_assignment represents a version identification for an external file.
description	type: text = string	OPTIONAL

Preprocessor Recommendations: The attribute name should be assigned a value of 'version' in this usage.

Postprocessor Recommendations: The name attribute value 'version' should be interpreted as an indication that the associated applied_identification_assignment represents simple version identification for the file.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#500=IDENTIFICATION_ROLE('version',$);
#510=DOCUMENT_FILE('doc id','',$,#520,'','');
#520=DOCUMENT_TYPE('unspecified type');
#530=APPLIED_IDENTIFICATION_ASSIGNMENT('THE VERSION ID',#500,(#510));
#540=DOCUMENT_REPRESENTATION_TYPE('digital',#510);
```

Example 28: exchange file for external file with version identification

6 Relationship Between Documents and Constituent Files

In 'Document as Product', the view definition is used to represent a definition of a particular document representation. There may be more than one representation definition associated with a document version. The document representation definition may be associated with the constituent external files that make it up. The association of constituent files with the definition of a document representation is optional.

If a file is under configuration control, it should be represented as a constituent of a document definition view/representation. In this case it is actually the managed document that is under direct configuration control, the file is in this way indirectly under configuration control. A change to the file results in a change to the managed document (i.e., a new version) - the changed file would be mapped as a constituent of a view/representation definition of the new document version. A simple external reference alone is not configuration controlled, it is just an external file reference to product data.

The AIM entity product_definition_with_associated_documents is used together with the identification of external files to support this requirement.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements of document identification with associated constituent external files are illustrated in Diagram 34.

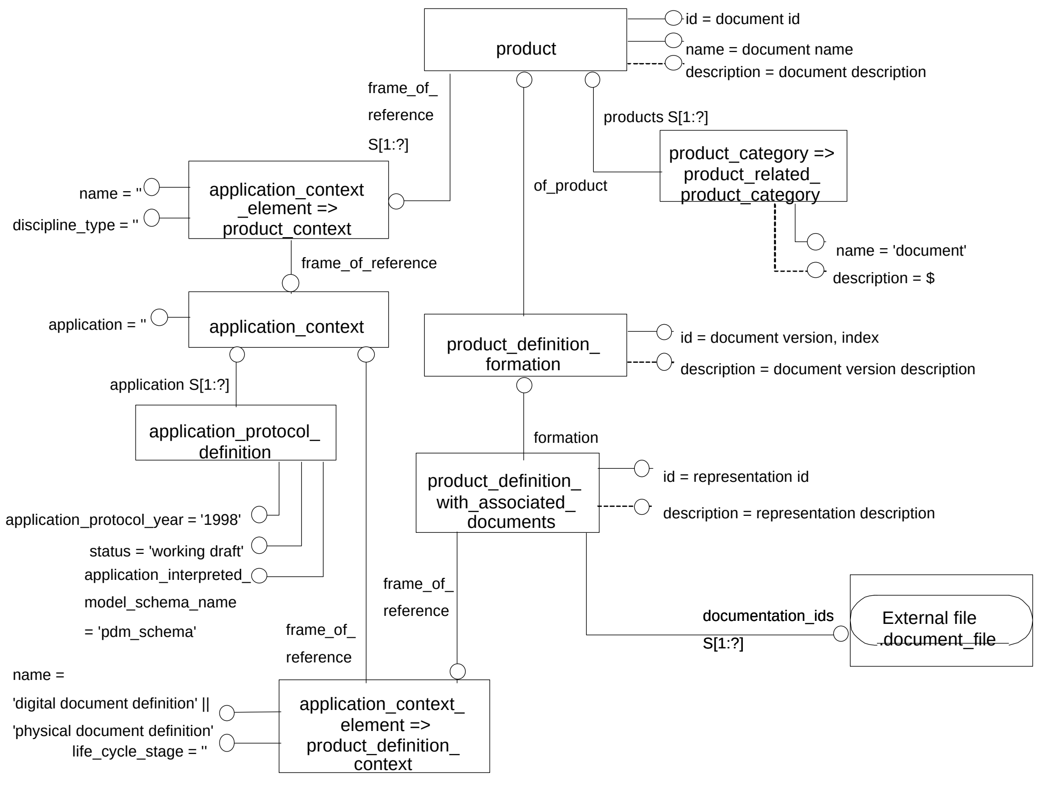

product
id = document id
name = document name
description = document description
frame_of_
reference
S[1:?]
products S[1:?]
product_category => product_related_ product_category
name = 'document'
description = $
of_product
name = ''
discipline_type = ''
application_context _element => product_context
frame_of_reference
application = ''
application_context
application S[1:?]
application_protocol_ definition
application_protocol_year = '1998'
status = 'working draft'
application_interpreted_
model_schema_name
= 'pdm_schema'
product_definition_ formation
id = document version, index
description = document version description
formation
product_definition_ with_associated_ documents
id = representation id
description = representation description
frame_of_
reference
frame_of_
reference
documentation_ids
S[1:?]
External file
document_file
name =
'digital document definition' ||
'physical document definition'
life_cycle_stage = ''
application_context_ element => product_definition_ context

Diagram 34: Document Master with Constituent External Files Instance Diagram

6.1.1.1 product_definition_with_associated_documents

The product_definition_with_associated_documents entity is a subtype of product_definition. It inherits the attributes id, description, formation and frame_of_reference from the supertype. This entity is only used in this case of 'Document as Product' and is not to be used in the case of 'Part as Product'. The attribute documentation_ids provides the relationship to the external file(s) that make up the actual content of the document representation definition.

Attributes

- The ***documentation_ids*** attribute contains a set of at least one instance representing the external file(s) that compose this view definition of the document.

ENTITY product_definition_with_ associated_documents	Attribute Population	Remarks
id	type: identifier = string	the identifier of this document view definition, should be unique in relation to a specific version
description	type: text = string	optional
formation	type: entity = product_definition_formation	References associated product_definition_formation
frame_of_reference	type: entity = product_definition_context	reference to the associated product_definition_context
documentation_ids	type: entity = document_file	SET[1:?] of files that make up the definition of the document

Preprocessor Recommendations: Preprocessors should use this entity instead of the supertype product_definition when relating the representation view definition of a 'Document as Product' to its constituent external files. The value of the attribute id should be unique in relation to a specific version.

Postprocessor Recommendations: Postprocessors should interpret this entity as a document view representation definition that identifies the constituent external files that make it up.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#270=PRODUCT('doc_4711','packaging guide','packaging guideline for
part',(#260));
#300=PRODUCT_RELATED_PRODUCT_CATEGORY('document',$,(#270));
#310=DOCUMENT_TYPE('unspecified');
#320=DOCUMENT_FILE('t1','text.doc','file with text for the
guide',#310,'','');
#330=PRODUCT_DEFINITION_FORMATION('ver3.2','version 3.2',#270);
#340=DOCUMENT_FILE('l1','',$,#310,'','');
#440=DOCUMENT_REPRESENTATION_TYPE('digital',#340);
#350=PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS('rep_id_3.2','represe
ntation of version 3.2',#330,#250,(#340,#320));
```

Example 29: exchange file for document with constituent external files

7 Document and File Association with Product Data

In 'Document as Product', documents and external files may be associated with product data. This association is done in a consistent way using an applied reference with a specified role. The applied reference is realized structurally by the PDM Schema entities document and applied_document_reference.

When associating a managed 'Document as Product' to product data, the document master is linked to the applied reference constructs using the additional entity document_product_equivalence. This linkage may be made at the level of the base identification, the document version, or the document representation view definition. The recommended level from which a document master should reference other product data is the document version.

External files may also be associated with product data, in a way that is structurally consistent with that used for documents, using the entity applied_document_reference. Since external files are represented by the entity document_file (a subtype of document), only the applied_document_reference entity is required to associate an external file as a reference to other product data.

7.1 Document Reference

In 'Document as Product', documents may be associated with product data by reference in a specified role. The base document identification, the document version, or the document representation definition may serve as the point of assignment for a document master to be associated with other product data. It is generally recommended to make a document reference from the level of document version.

Reference of a managed document to product data is accomplished using the entities document_product_equivalence, document, document_type, and applied_document_reference.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements of document association with other product data are shown in Diagram 35.

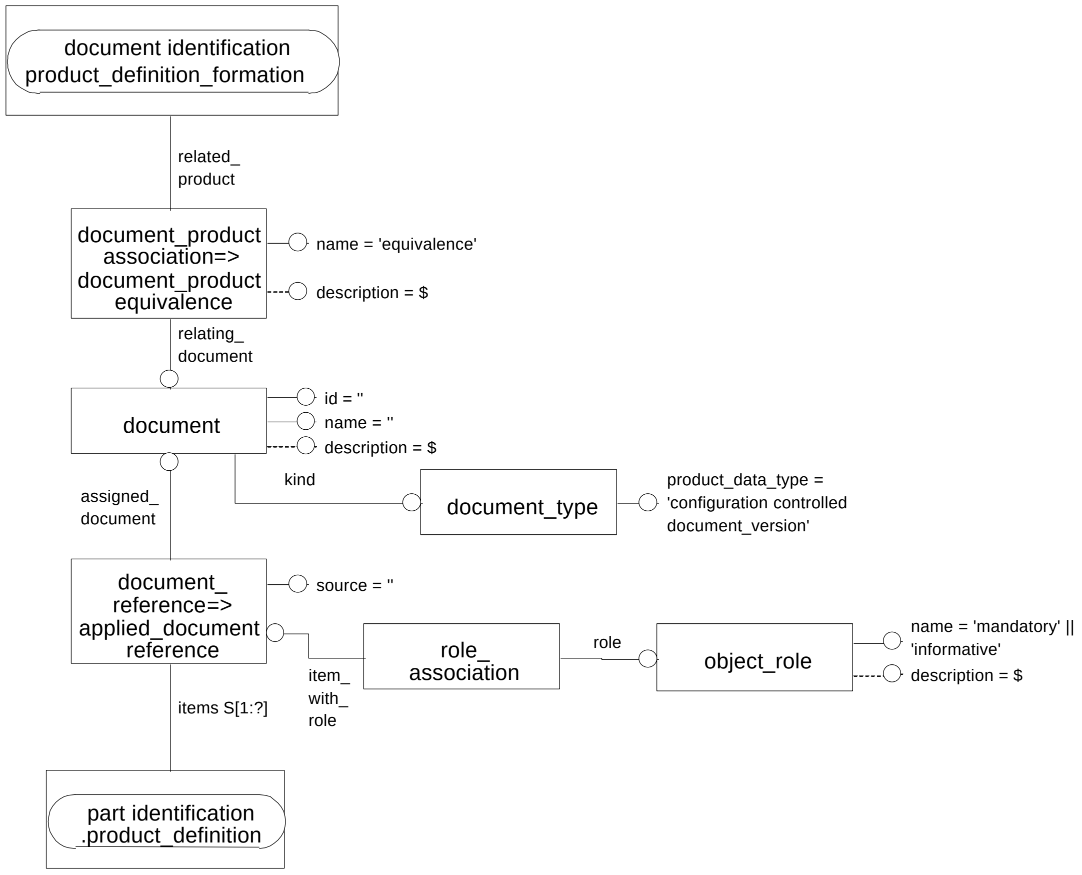


Diagram 35: Document Association to Product Data Instance Diagram

7.1.1.1 document_product_equivalence

This entity is a subtype of document_product_association relating a document to a 'product'. It asserts that the related product (or product_definition_formation or product_definition) in this case represents an element of a document master that is assigned as a reference to some other product data. The related_product attribute may refer to an instance of the entity product, product_definition_formation, or product_definition by way of the select type product_or_definition_or_formation. This provides the possibility to assign it to some other product data as a reference when used with the related entities document and applied_document_reference.

Attributes

- The ***name*** attribute characterizes the nature of this relationship.
- The ***description*** attribute is optional.
- The ***related_product*** attribute references the portion of the document master that is being assigned.
- The ***relating_document*** attribute has a document entity in a document_reference to some product data.

ENTITY document_product_equivalence	Attribute Population	Remarks
name	type: label = string 'equivalence'	
description	type: text = string	OPTIONAL
related_product	type:	specifies the component of the

ENTITY document_product_equivalence	Attribute Population	Remarks
	product_or_definition_or_formation (select)	document master that is the point of association to product data
relating_document	type: entity = document	

Preprocessor Recommendations: This entity should be used only when the requirement exists to associate a 'Document as Product' with product data. If the 'Document as Product' is not associated with product data, then this entity should not be instantiated.

The relating_document attribute always has a document entity as its value. Additional constraints apply to the document_type entity related to this document by the kind attribute. These prescribe a value for the product_data_type attribute of this document_type entity, correlated with the following select type values of the relating_document :

• Product	=> 'configuration_controlled_document'
• Product_definition_formation	=> 'configuration_controlled_document_version'
• Product_definition	=> 'configuration_controlled_document_definition'

Of these possible options for assignment, it is recommended to assign the 'configuration controlled document version' to product data. When assigned as a reference to a part master identification, it is recommended to assign the document version (product_definition_formation) to the view definition (product_definition) within the part master.

Postprocessor Recommendations: When importing an instance of this entity, postprocessors should recognize that the related_product attribute references a 'Document as Product'. The relating_document attribute references an instance of document that exists to allow assignment of the 'Document as Product' to product data.

Related Entities: There are no specific related entities.

7.1.1.2 Document

This entity is only instantiated as itself when used as the relating_document in a document_product_equivalence relationship. In this role, the document entity is equated with the document master, and works together with the applied_document_reference to support the assignment of managed documents to product data in the STEP PDM Schema.

Attributes

- The ***kind*** attribute references a document_type entity.

ENTITY document	Attribute Population	Remarks
id	type: identifier = string	Not to be used for valid user data.
name	type: label = string	
description	type: text = string	OPTIONAL
kind	type: entity = document_type	The product_data_type attribute on this referenced entity is constrained by the entity document_product_equivalence

Preprocessor Recommendations: The document entity is only to be instantiated as itself in this role within the document_product_equivalence. In this case, the document identification is represented by the product master - the attributes on the document entity should not be used for identification of the managed document. It is not instantiated as itself, but as the subtype document_file when used to represent the identification of an external file reference (see 5.1).

Postprocessor Recommendations: The document entity is only a structural element in the assignment of a managed document to product data, along with the entity applied_document_reference. Identification of the managed document is represented by the product master entities according to the 'Document as Product' approach.

Related Entities: There are no specific related entities.

7.1.1.3 document_type

This entity is required for each instance of the entity document. In the given context, this entity is used to indicate that the related document objects are under configuration control. This corresponding usage of document_type is not to be used for a specific document object classification.

Attributes

- The ***product_data_type*** attribute describes the type of product data represented by the document entity.

ENTITY document_type	Attribute Population	Remarks
product_data_type	type: identifier = string	

Preprocessor Recommendations: In this use for document master identification, the attribute product_data_type is constrained by the entity document_product_equivalence to take one of the following values : 'configuration_controlled_document', 'configuration_controlled_document_version', or 'configuration_controlled_document_definition'.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

7.1.1.4 applied_document_reference

This entity is a subtype of document_reference. It supports the assignment of documents to elements of product data within the select type document_reference_item. Notably, documents may be assigned to the part master view definition (product_definition) or to an individual occurrence of a part definition usage in an assembly structure (product_definition_relationship).

Attributes

- The ***source*** attribute is inherited from the supertype document_reference.
- The ***items*** attribute indicates the elements of product data referenced by the managed document.

ENTITY applied_document_reference	Attribute Population	Remarks
source	type: label = text	
assigned_document	type: entity = document	This referenced instance is the only case where the document entity is instantiated as itself, and not the subtype document_file.
items	type: document_reference_item = select	SET [1:?]

Preprocessor Recommendations: The inverse attribute role requires the associated entities role_association and object_role be instantiated related to this entity. Preprocessors should support as a minimum the assignment of document to the product_definition representing a part view definition.

Postprocessor Recommendations: Postprocessors should as a minimum recognize the assignment of document to the product_definition representing a part view definition.

Related Entities: There are no specific related entities.

7.1.1.5 role_association

This entity provides the item_with_role with a named role string. This is the method to add a role attribute to referenced entities that do not have one defined.

Attributes

- The ***item_with_role*** attribute.
- The ***role*** attribute references an instance of the entity object_role.

ENTITY role_association	Attribute Population	Remarks
item_with_role	type: role_select = select	
role	type: entity = object_role	

Preprocessor Recommendations: There are no specific preprocessor recommendations.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

7.1.1.6 object_role

This entity assigns a role to the associated document reference. Two roles are defined:

- 'mandatory' means the assignment of the document to the product data is in a mandatory relationship **-** the document must be taken into account to understand the complete product information.
- 'informative' means the assignment of the document to the product data is an optional relationship **-** the document may be considered for additional information but is not required or enforced.

Attributes

- The ***name*** attribute indicates the name of the role.
- The ***description*** attribute is optional.

ENTITY object_role	Attribute Population	Remarks
name	type: label = string 'mandatory', 'informative'	
description	type: text = string	OPTIONAL

Preprocessor Recommendations: The name attribute should have the value 'mandatory' or 'informative' to indicate if the associated document reference is required or optional.

Postprocessor Recommendations: The value of the name attribute shall indicate if the document assignment is mandatory or optional for information only.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#70=PRODUCT('MP-03-2','my part',$,(#60));
#80=PRODUCT_DEFINITION_FORMATION('03','3rd modification',#70);
#90=PRODUCT_DEFINITION('/NULL',$,#80,#50);
...
#360=DOCUMENT_PRODUCT_EQUIVALENCE('equivalence',$,#370,#280);
#370=DOCUMENT('','',$,#380);
#380=DOCUMENT_TYPE('configuration controlled document version');
#390=APPLIED_DOCUMENT_REFERENCE(#370,'',(#90));
#400=ROLE_ASSOCIATION(#410,#390);
#410=OBJECT_ROLE('mandatory',$);
```

Example 30: exchange file segment for document association to product data

7.2 External File Reference

External files may also be associated with product data in a way that is consistent but simpler than that used for documents. While the requirement to assign documents to various product data is basically supported by three additional entities: document_product_equivalence, document, and applied_document_reference, external files need only the applied_document_reference entity to be related as a reference to other product data.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of external file association to product data are illustrated in Diagram 36.

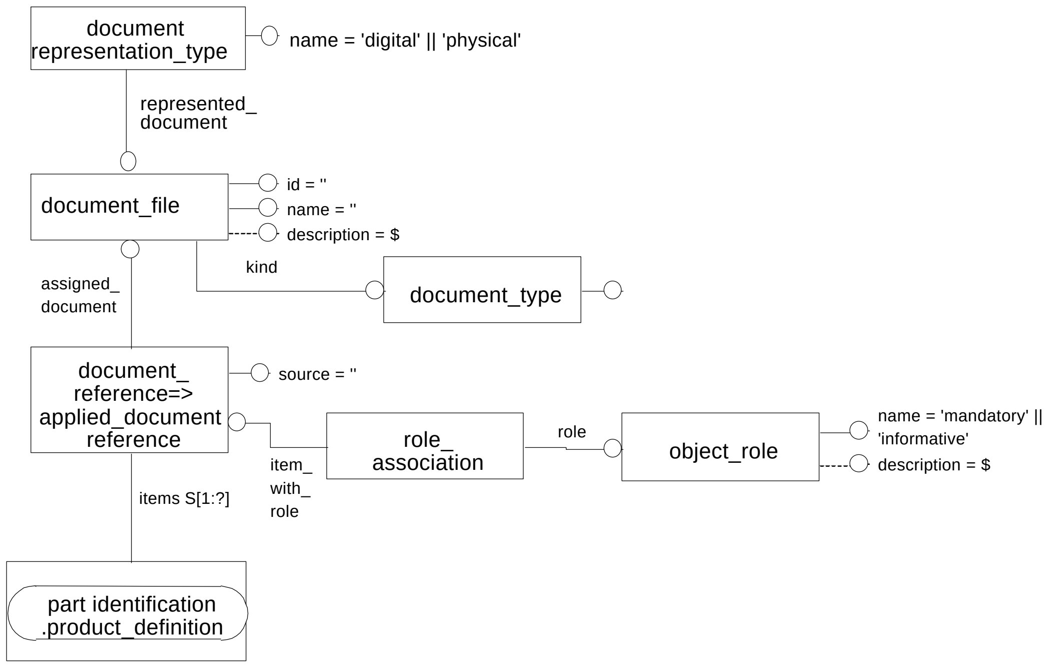


Diagram 36: External File Association to Product Data Instance Diagram

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#70=PRODUCT('MP-03-2','my part',$,(#60));
#80=PRODUCT_DEFINITION_FORMATION('03','3rd modification',#70);
#90=PRODUCT_DEFINITION('/NULL',$,#80,#50);
...
#120=DOCUMENT_TYPE('geometry');
#130=DOCUMENT_FILE('measure file id','measure data',$,#120,'','');
#140=APPLIED_DOCUMENT_REFERENCE(#130,'',(#90));
#150=OBJECT_ROLE('informative',$);
#160=ROLE_ASSOCIATION(#150,#140);
#170=DOCUMENT_REPRESENTATION_TYPE('digital',#130);
```

Example 31: exchange file segment for external file association to product data

7.3 Constrained Document or File Reference

Constraints may be specified on the association of documents or files with product data, in order to distinguish a portion of the entire document or file that applies in the reference.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of constrained document or file association to product data are illustrated in Diagram 37.

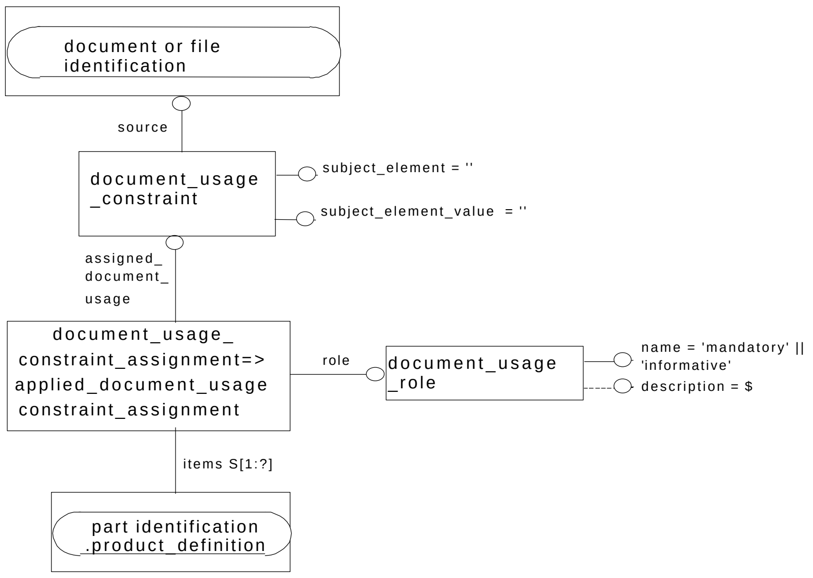

document or file identification
source
document_usage _constraint
subject_element = ''
subject_element_value = ''
assigned_ document_ usage
document_usage_ constraint_assignment=> applied_document_usage constraint_assignment
role
document_usage _role
name = 'mandatory' || 'informative'
description = $
items S[1:?]
part identification .product_definition

Diagram 37: Constrained Document Association to Product Data Instance Diagram

7.3.1.1 document_usage_constraint

This entity identifies a portion of a document that is applicable for a given usage.

Attributes

- The ***source*** attribute references the complete document or file of which a portion is distinguished.
- The ***subject_element*** attribute describes the portion or element of the complete document.
- The ***subject_element_value*** attribute conveys a specific value of the subject_element.

ENTITY document_usage_constraint	Attribute Population	Remarks
source	type: entity = document	May be document_file in the case of an external file reference
subject_element	type: label = string	
subject_element_value	type: text = string	

Preprocessor Recommendations: The subject_element should identify the relevant portion or section of the document that is applicable. The subject_element_value describes how the subject_element is to be interpreted.

Postprocessor Recommendations: Postprocessors should interpret the subject_element as an indication of the relevant portion of the entire document.

Related Entities: There are no specific related entities.

7.3.1.2 applied_document_usage_constraint_assignment

This entity is a subtype of document_usage_constraint_assignment. It supports the constrained assignment of a document to product data. A constrained assignment identifies only a portion of the entire document that is relevant in the assignment.

Attributes

- The ***assigned_document_usage*** attribute references the associated document_usage_constraint.
- The ***role*** attribute identifies the role of the assignment.
- The ***items*** attribute references the product data to which the partial document is assigned.

ENTITY applied_document_usage_constraint_assignment	Attribute Population	Remarks
assigned_document_usage	type: entity = document_usage_constraint	
role	type: entity = document_role	
items	type : docuement_reference_item = select	

Preprocessor Recommendations: There are no specific preprocessor recommendations.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

7.3.1.3 document_usage_role

This entity identifies the role of the constrained document assignment.

Attributes

- The ***name*** attribute identifies the role of the document assignment.
- The ***description*** attribute is optional.

ENTITY document_usage_role	Attribute Population	Remarks
name	type : label = string 'mandatory', 'informative'	
description	type : text = string	OPTIONAL

Preprocessor Recommendations: The name attribute should contain the value 'mandatory' in case the document assignment is required, or 'informative' if the document assignment is optional.

Postprocessor Recommendations: Postprocessors should interpret the name attribute as an indication of whether the document assignment is required ('mandatory') or optional ('informative').

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#180=DOCUMENT_TYPE('geometry');
#190=DOCUMENT_FILE('plot_1','plot data',$,#180,'','');
#200=DOCUMENT_USAGE_CONSTRAINT(#190,'sheet','one');
#210=DOCUMENT_USAGE_ROLE('informative',$);
#220=APPLIED_DOCUMENT_USAGE_CONSTRAINT_ASSIGNMENT(#200,#210,(#90));
#230=DOCUMENT_REPRESENTATION_TYPE('physical',#190);
```

Example 32: exchange file segment for constrained document association to product data

The following section combines the above discussed concepts and segments in a complete example. The concepts instantiated in the example are illustrated in Figure 10.

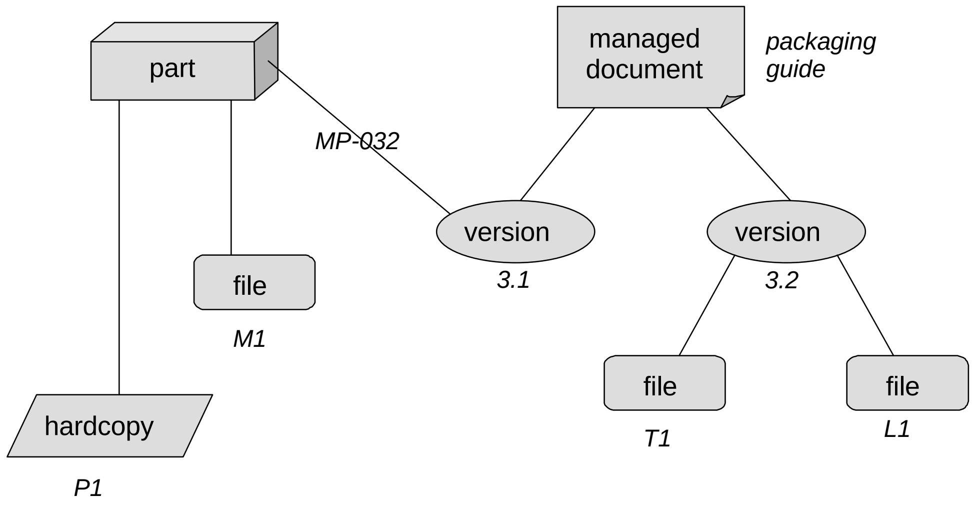


Figure 10: Schematic Overview of Complete Document and File Example

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements of constrained document or file association to product data are illustrated in Diagram 38, Diagram 39, Diagram 40, and Diagram 41.

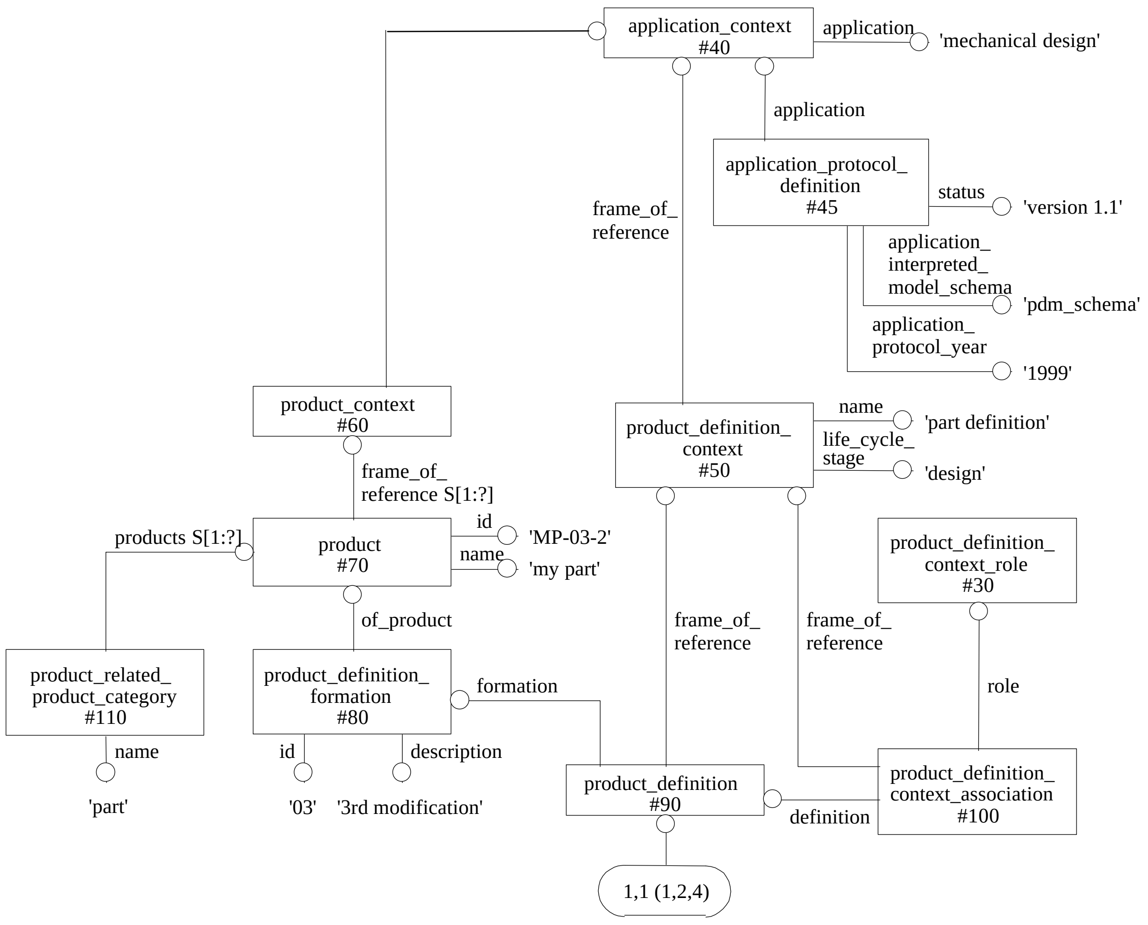


Diagram 38 : Part Master Instance Diagram

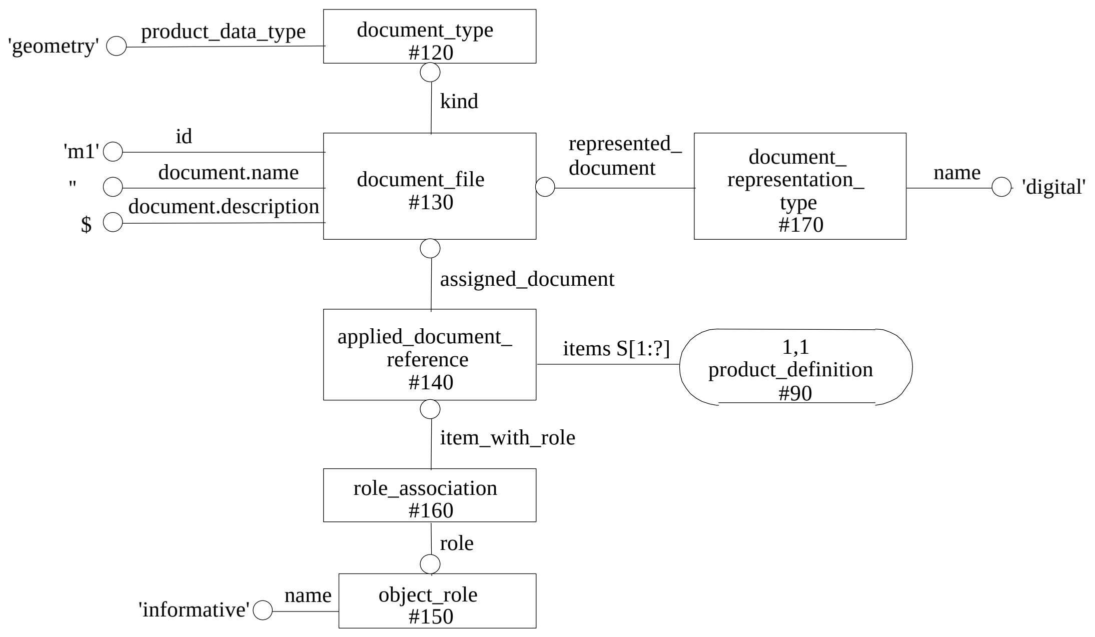


Diagram 39: External File Reference to Part View Definition Instance Diagram

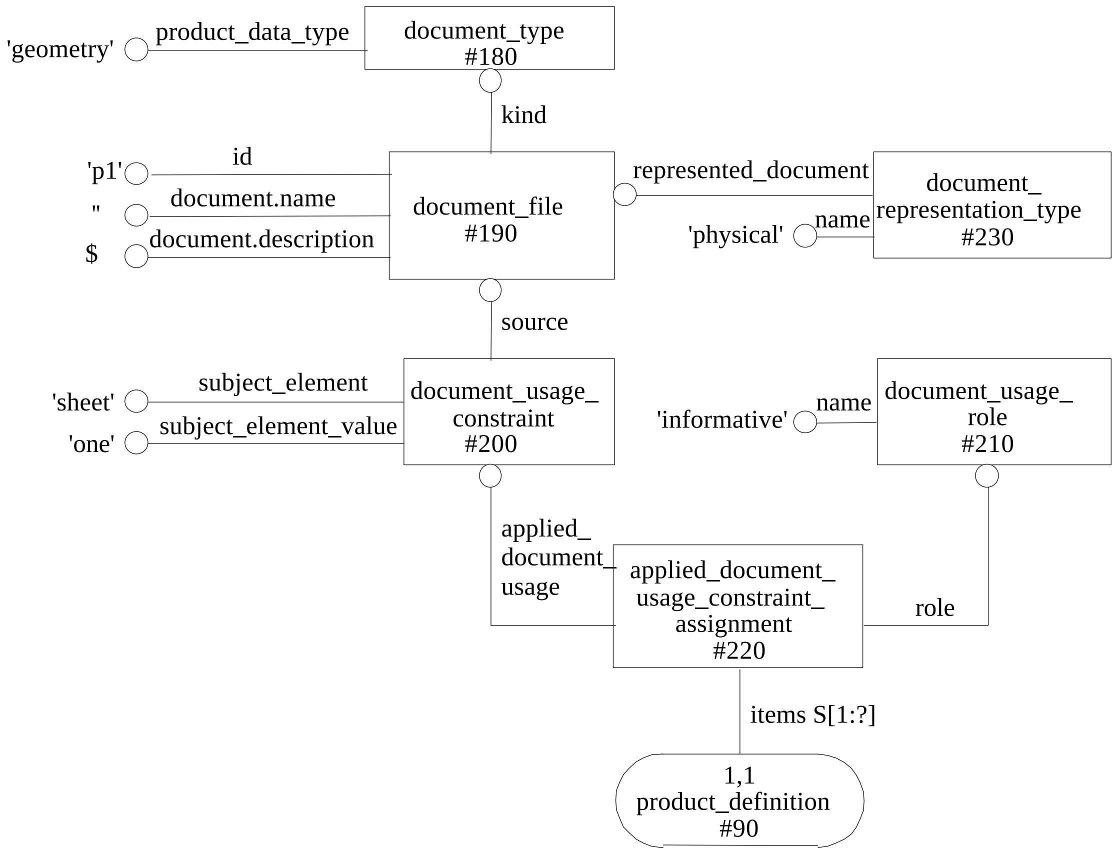


Diagram 40: Constrained External File Reference to Part View Definition Instance Diagram

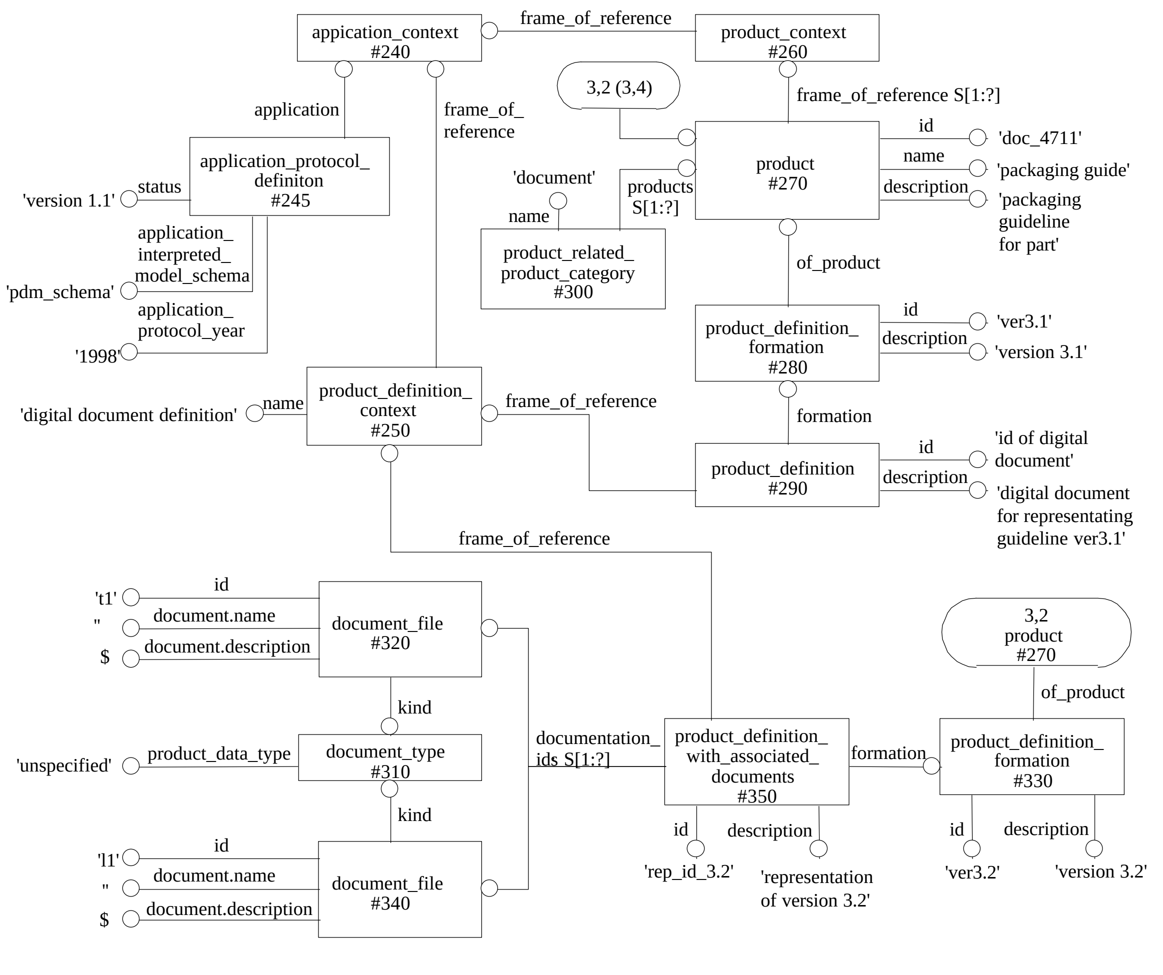


Diagram 41: Document and Constituent File Association Instance Diagram

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('pdm_schema data', 'pdm_schema'), '2;1');
FILE_NAME('C:document_xample.stp', '1999-5-20 T12:39:1', (''), (), '',
   '', '');
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;

DATA;
/*  Entities #10 - #20 define part master data                         */
/*  To the product_definition #90 is a life cycle specific view on     */
/*  that part to which documents and external files                    */
/*  are assigned in the following sections of this file                */
#30=PRODUCT_DEFINITION_CONTEXT_ROLE('',$);
#40=APPLICATION_CONTEXT('mechanical design');
#45=APPLICATION_PROTOCOL_DEFINITION('version
1.1','pdm_schema',1998,#40);
#50=PRODUCT_DEFINITION_CONTEXT('part definition',#40,'design');
#60=PRODUCT_CONTEXT('',#40,'');
```

```
#70=PRODUCT('MP-03-2','my part',$,(#60));
#80=PRODUCT_DEFINITION_FORMATION('03','3rd modification',#70);
#90=PRODUCT_DEFINITION('/NULL',$,#80,#50);
#100=PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#90,#50,#30);
#110=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#70));

/*  Entities #120 to #170 model the assignment of                   */
/* digital file data to the product_definition #90                  */
#120=DOCUMENT_TYPE('geometry');
#130=DOCUMENT_FILE('m1','',$,#120,'','');
#140=APPLIED_DOCUMENT_REFERENCE(#130,'',(#90));
#150=OBJECT_ROLE('informative',$);
#160=ROLE_ASSOCIATION(#150,#140);
#170=DOCUMENT_REPRESENTATION_TYPE('digital',#130);

/* Entities #180 to #230 model the assignment of hardcopy document
*/
/* data to the product_definition #90                                */
/*  This assignment is partial, i.e., in addition the information is */
/*  conveyed that only sheet one of the hardcopy (plots) shall be    */
/*  logically assigned                                               */
#180=DOCUMENT_TYPE('geometry');
#190=DOCUMENT_FILE('p1','',$,#180,'','');
#200=DOCUMENT_USAGE_CONSTRAINT(#190,'sheet','one');
#210=DOCUMENT_USAGE_ROLE('informative',$);
#220=APPLIED_DOCUMENT_USAGE_CONSTRAINT_ASSIGNMENT(#200,#210,(#90));
#230=DOCUMENT_REPRESENTATION_TYPE('physical',#190);

/* Entities #240 to #350 model a managed document (document       */
/* as product)  For this document two versions are specified      */
/*(#280 and #330)To version 3.2 (#330) two files (#320, #340)     */
/* that constitute the document are defined                       */
#240=APPLICATION_CONTEXT('');
#250=PRODUCT_DEFINITION_CONTEXT('digital document definition',#240,'');
#260=PRODUCT_CONTEXT('',#240,'');
#270=PRODUCT('doc_4711','packaging guide','packaging guideline for
part',(#260));
#280=PRODUCT_DEFINITION_FORMATION('ver3.1','version 3.1',#270);
#290=PRODUCT_DEFINITION('id of digital document','digital document for
representing guideline ver3.1',#280,#250);
#300=PRODUCT_RELATED_PRODUCT_CATEGORY('document',$,(#270));
#310=DOCUMENT_TYPE('unspecified');
#320=DOCUMENT_FILE('t1','',$,#310,' ',' ');
#330=PRODUCT_DEFINITION_FORMATION('ver3.2','version 3.2',#270);
#340=DOCUMENT_FILE('l1','','file with logo for the guide',#310,'','');
#350=PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS('rep_id_3.2','represe
ntation of version 3.2',#330,#250,(#340,#320));

/* Entities #360 to #410 establish the association of the            */
/* managed document version #280 (packaging guide) with the view     */
/* of the part modeled via product_definition #90                    */
#360=DOCUMENT_PRODUCT_EQUIVALENCE('equivalence',$,#370,#280);
#370=DOCUMENT('','',$,#380);
#380=DOCUMENT_TYPE('configuration controlled document version');
#390=APPLIED_DOCUMENT_REFERENCE(#370,'',(#90));
#400=ROLE_ASSOCIATION(#410,#390);
#410=OBJECT_ROLE('mandatory',$);
```

```
ENDSEC;
END-ISO-10303-21;
```

Example 33: exchange file for complete document and file identification and association

8 Document Properties

In the PDM Schema, document properties can be associated with representations of documents as well as to individual files. If document properties are assigned to representations of documents the characteristics apply as well to all the constituent files of the document representation in most cases. Some properties with numeric values, such as 'file size' and 'page count', applied to the document representation will not correspond to an individual file in a multiple file document representation, but to the sum of all the files that make up the particular document representation definition. To avoid redundancy it is recommended that properties that are shared by all constituents of a given document representation be directly associated with that document representation rather than replicated with the individual files.

The general schema for the assignment of properties to document representation (i.e. product_definition) or document files (i.e. document_file) is to instantiate a property definition tree with property_definition, property_definition_representation, representation and eventually multiple instantiations of descriptive_representation_item (or measure_representation_item) that provide the representation of a specific aspect of the represented property.

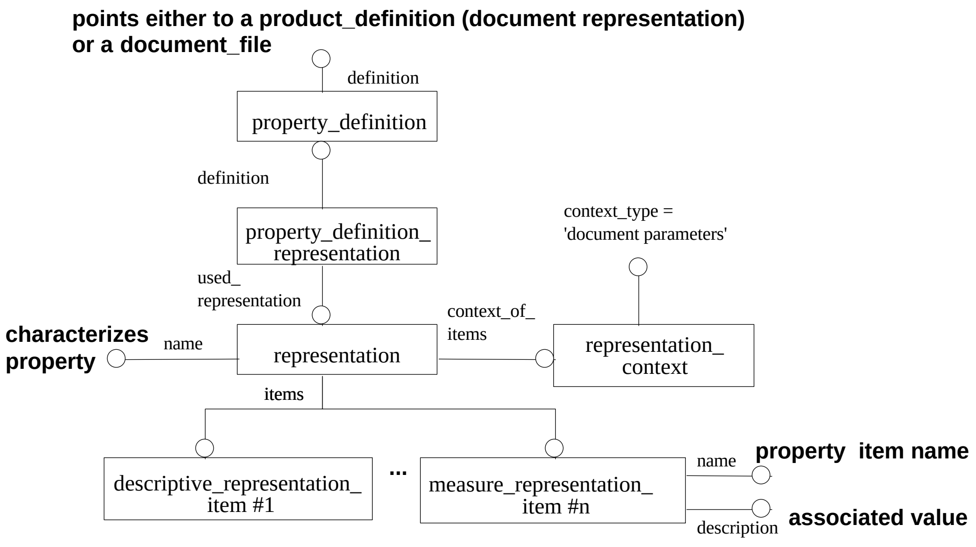

points either to a product_definition (document representation) or a document_file
definition
property_definition
definition
property_definition_ representation
context_type = 'document parameters'
used_ representation
context_of_ items
characterizes property
name
representation
representation_ context
items
descriptive_representation_ item #1
...
measure_representation_ item #n
name
property item name
description
associated value

Diagram 42: General pattern for the association of document properties

8.1.1.1 property_definition

A property_definition is in the given context a property that characterizes a document representation or document file. It applies either to a product_definition (for the document representation) or a document_file.

Attributes

- The *name* attribute indicates via 'document property' that the property is related to a document representation or a document file.
- The *description* attribute provides additional description of the property.
- The *definition* attribute references the document object which is characterized by the property_definition.

ENTITY property_definition	Attribute Population	Remarks
name	type: identifier = string	should be instantiated as 'document property' in the given context
description	type: text = string	optional, shall not be instantiated
definition	type: entity = product_definition or product_definition_with_ associated_documents or document_file	References associated document

Pre-processor Recommendations: It is recommended to instantiate no more than one property_definition with the value of the name attribute equal to 'document property' for each document representation definition (product_definition or product_definition_with_associated_documents)or document_file. One property_definition per document object shall be used to collect all document object properties via associated property_definition_representations.

Post-processor Recommendations: none specific

Related Entities: none specific

8.1.1.2 property_definition_representation

A property_definition_representation is in the given context an association between a document property and its representation.

Attributes

- The *definition* attribute references the property_definition that is defined by the associated representation.
- The *used_representation* attribute points to a representation that collects the representations items that together describe the property.

ENTITY property_definition_representation	Attribute Population	Remarks
definition	type: entity = property_definition	References associated property_definition
used_representation	type: entity = representation	References associated representation

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

8.1.1.3 Representation

A representation is in the context of document properties a collection of one or more descriptive_representations_items that are related in a representation_context with type 'document parameters'. Herein a representation represents a document property.

Attributes

- The *name* attribute characterizes the type of the document related property via a string.

- The *items* attribute collect items that represent the values for a given property instantiation.
- The *context_of_items* attribute points to a representation_context. The representation_context has the type 'document parameters'.

ENTITY representation	Attribute Population	Remarks
name	type: label = STRING	the name attribute characterizes the document related property
items	type: entity = descriptive_representation_item	the items that constitute the representation of the document property
context_of_items	type: entity = representation_context	is required to point to a representation_context with representation_context.type set to 'document parameters'
id	derived	should not be instantiated
description	derived	should not be instantiated

Pre-processor Recommendations: The name characterizes the document related property. The representation is required to have a context with representation_context.type = 'document parameters'.
For a given document representation or document_file, the name of all associated representation objects shall be different for the properties described in sections 1.2 through 1.5.

Post-processor Recommendations: A post-processor shall in the given context support the following values for representation.name 'document content', 'document creation', 'document format', 'document size'.

Related Entities: none specific

8.1.1.4 descriptive_representation_item

A descriptive_representation_item is in the document property context a textual element that participates in one or more representations to define the respective properties.

Attributes

- The *name* attribute characterizes the information modeled with the descriptive_representation_item via a string.
- The *description* attribute defines a textual value as an instantiation of the modeled property.

ENTITY descriptive_representation_item	Attribute Population	Remarks
name	type: label = STRING	the name attribute indicates the name of the represented property
description	type: text = STRING	the description is the value associated with the representation item in textual form

Pre-processor Recommendations: none specific

Post-processor Recommendations: In the given context the following instantiations of descriptive_representation_item.name shall be expected: 'detail level', 'geometry', 'language', 'real world scale', 'creating interface', 'creating system', 'operating system', 'data format', 'character code', 'size format', 'size format standard'.

Related Entities: The descriptive_representation_items for a given document property are collected in a representation. The representation characterizes the property via the attribute representation.name.

8.2 Document content property

The content related properties of documents are represented accordingly to the general scheme outlined above. To indicate that the property is content related the used representation must be instantiated with:

representation.name = 'document content'

This document content capability specifies characteristics explaining precisely the content of a given document object. Aspects of the content property can be modeled via the following instantiations of descriptive_representation_items collected in the above representation.

requirement	descriptive_-representation_item.name	descriptive_representation_item.description
The level of detail that the document file or the document representation provides.	'detail level'	Where applicable the following values shall be used: • 'rough 3d shape': 3D shape model without edge rounds and fillets; • 'rounded edges': 3D shape model with edge rounds and fillets.
The kind or kinds of geometry that an object contains	'geometry'	Where applicable the following values shall be used: • '3D wireframe model': The document contains a 3D shape model in wireframe representation; • '2D shape': The document contains a 2D shape model or contours only; • 'surface model': The document contains a 3D shape model in surface representation; • 'closed volume': The document contains a 3D shape model in closed body topological surface representation; • 'solid model': The document contains a 3D shape model in advanced boundary representation; • 'solid and surface model': The document contains a 3D shape model in surface and advanced boundary representation; • 'assembly': The document contains an assembly structure with reference to the assembled components and their transformation matrices; • 'assembly with mating elements': The document contains an assembly structure including the mating components only, such as screws or rivets, with exact positioning information. This assembly representation is intended to be overlaid with the assembly structure for the main components; • '2D drawing': The document contains a technical drawing without 3D shape representation; • 'drawing derived from 3D data': The document contains a technical drawing that has been derived from a 3D shape model; • 'drawing related to 3D data': The document contains a technical drawing that visualizes a 3D shape model and possibly establishes associative links to the 3D shape model.
language or	'language'	e.g. 'English"

languages are used in the characterized objects.		
the scale that is used	'real world' scale'	e.g. '1:50'

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* Entities #210 to #250 assign document content parameters to 'file1' (#80)
#210 = PROPERTY_DEFINITION('document property', '', #80);
#220 = PROPERTY_DEFINITION_REPRESENTATION(#210, #230);
#230 = REPRESENTATION('document content', (#240, #250), #200);
#240 = DESCRIPTIVE_REPRESENTATION_ITEM('detail level', 'rough 3D shape');
#250 = DESCRIPTIVE_REPRESENTATION_ITEM('geometry type', 'solid model');
```

Example 34: exchange file segment for content property representation

8.3 Document creation property

The content related properties of documents are represented accordingly to the general scheme outlined above.
To indicate that the property is related to document creation the used representation must be instantiated with:

representation.name = 'document creation'

Aspects of the creation property can be modeled via the following instantiations of descriptive_representation_items collected in the above representation.
It is recommended that when document creation property information is represented in a corresponding representation, at least a descriptive_representation_item with name 'creating system' shall be instantiated.

requirement	**descriptive_-representation_item.name**	**descriptive_representation_item.description**
the computer application used to create the document object.	'creating interface'	e.g. 'Postscript driver'
the computer application or the machine which is used to create the object that is characterized.	'creating system'	e.g. 'Microsoft Word'
the operating system that is used to execute the computer application that created the characterized object.	'operating system'	e.g. 'HP-UX 11'

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* Entities #210 to #250 assign document creation parameters to 'file1' (#80)*/
#260 = PROPERTY_DEFINITION('document property', '', #80);
#270 = PROPERTY_DEFINITION_REPRESENTATION(#260, #280);
#280 = REPRESENTATION('document creation', (#290, #300, #310), #200);
#290 = DESCRIPTIVE_REPRESENTATION_ITEM('creating system', 'My CAD');
#300 = DESCRIPTIVE_REPRESENTATION_ITEM('operating system', 'Linux 2.1');
#310 = DESCRIPTIVE_REPRESENTATION_ITEM('creating interface',
   'export driver');
```

Example 35: exchange file segment for creation property representation

8.4 Document format property

The format for related properties of documents are represented accordingly to the general scheme outlined above.
To indicate that the property is related to document format the used representation must be instantiated with:

representation.name = 'document format'

This capability specifies the format of a document object. At least one of the items shall be specified for each instance of this property representation.
Aspects of the format property can be modeled via the following instantiations of descriptive_representation_items collected in the above representation.

requirement	**descriptive_-representation_item.name**	**descriptive_representation_item.description**
the convention that was used to structure the information in the characterized object	'data format'	where applicable the following values shall be used: 'DXF', 'IGES', 'STEP AP203', 'STEP AP214', 'TIFF CCITT GR4', 'VDAFS', 'VOXEL'
the character code that is used for the stored data	'character code'	where applicable the following values shall be used: 'US ASCII 7bit', 'ISO LATIN-1', 'EBCDIC', 'binary'
the dimensions of a physical presentation	'size format' or 'size format standard' where applicable	e.g. 'ISO A0', '0.2 x 0.4 x 0.4 meters'

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
/* Entities #140 to #200 assign document format parameters to 'file1' (#80)
#140 = PROPERTY_DEFINITION('document property', '', #80);
#150 = PROPERTY_DEFINITION_REPRESENTATION(#140, #160);
#160 = REPRESENTATION('document format', (#170, #180, #190), #200);
#170 = DESCRIPTIVE_REPRESENTATION_ITEM('document format', 'VOXEL');
#190 = DESCRIPTIVE_REPRESENTATION_ITEM('character code', 'binary');
#200 = REPRESENTATION_CONTEXT('', 'document parameters');
```

Example 36: exchange file segment for format property representation

8.5 Document size property

The document size related properties of documents are represented accordingly to the general scheme outlined above.
To indicate that the property is content related the used representation must be instantiated with:

representation.name = 'document size'

This capability specifies the size of document object. At least one of the items shall be specified for each instance of this property representation.
The size property capability differs from the general scheme in the fact that measure_representation_items are instantiated instead of descriptive_representation_items. Aspects of the size property can be modeled via the following instantiations of measure_representation_items collected in the representation.

requirement	**measure_representation_item.name**
the value that represents the size of a digitally stored document.	'file size'
the number of pages (of a physical document)	'page count'

8.5.1.1 measure_representation_item

A descriptive_representation_item is in the document property context an element that participates in one or more representations to define the respective properties by defining measure values with associated units.

Attributes

- The *name* attribute characterizes the information modeled with the measure_representation_item via a string.
- The *value_component* attribute defines a value as an instantiation of the modeled property.
- The *unit_component* attribute provides the unit for the size or page count measure.

ENTITY measure_representation_item	Attribute Population	Remarks
name	type: label = STRING	must be 'file size' or 'page count' in the given context
value_component	type: select measure_value = REAL | -NUMBER	to indicate the select type it is recommended to instantiate COUNT_MEASURE(nr) for page numbers and CONTEXT_DEPENDENT_MEASURE(size) for 'file size'
unit_component	type: select unit = NAMED_UNIT | DERIVED_UNIT	It is recommended to use a context_dependent_unit as unit component

Pre-processor Recommendations: For 'page count' it is recommended to instantiate an INTEGER as value_component. For 'file size' it is recommended to instantiate a REAL. The related dimensional_ exponents should all be set to 0.

Post-processor Recommendations: none specific

Related Entities: none specific

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#90 = DOCUMENT_TYPE('');
#100 = DOCUMENT_FILE('hardcopy', '', $, #90, '', '');
#110 = DOCUMENT_REPRESENTATION_TYPE('physical', #100);
#120 = PROPERTY_DEFINITION('document property', '', #100);
#130 = PROPERTY_DEFINITION_REPRESENTATION(#120, #140);
#140 = REPRESENTATION('document size', (#150), #200);
#150 = MEASURE_REPRESENTATION_ITEM('page count', COUNT_MEASURE(1), #170);
#160 = DIMENSIONAL_EXPONENTS(0.0E+000, 0.0E+000, 0.0E+000, 0.0E+000,
   0.0E+000, 0.0E+000, 0.0E+000);
#170 = CONTEXT_DEPENDENT_UNIT('pages',#160);
```

Example 37: exchange file segment for size property representation

8.6 Document source property

This capability allows specification where a document object can be found in a digital or physical data storage system. This capability differs in its instantiation from the general pattern. The instantiation diagram is shown below.

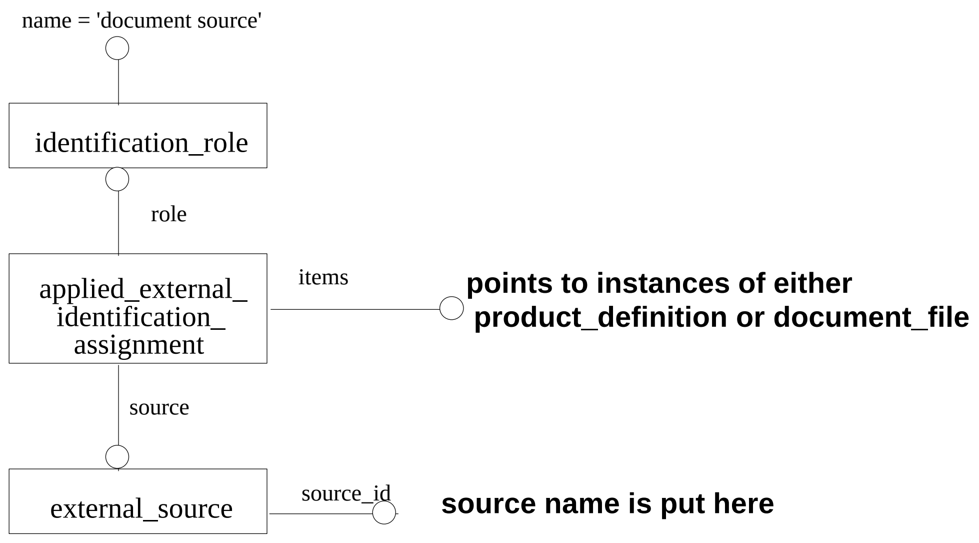

name = 'document source'
identification_role
role
applied_external_ identification_ assignment
items
points to instances of either product_definition or document_file
source
external_source
source_id
source name is put here

Diagram 43 : Document Source Property Instance Diagram

8.6.1.1 identification_role

Identification role provides a name and an optional description for an identification assignment. In the context of this capability the entity is used to specify that the assignment structure is used to represent the document source property.

Attributes

- The *name* attribute characterizes the role of the related identifcation_assignment as 'document source'.
- The *description* attribute provides optional textual information on the identification_role.

ENTITY identification_role	Attribute Population	Remarks
name	type: label = STRING	must be 'document source' for this capability
description	type: text = STRING	should not be instantiated

Pre-processor Recommendations: Identification_role.name shall be instantiated as 'document source' in the given context. Identifcation_role.description is an optional attribute and shall not be instantiated here.

Post-processor Recommendations: none specific

Related Entities: none specific

8.6.1.2 applied_external_identification_assignment

Applied_external_identification_assignment is used to assign an external source and identifier to a set of items. In the context of the document source capability it assigns the location name given via external_source.id to document objects.

Attributes

- The *assigned_id* attribute allows for an identification associated with the referenced document data. It is recommended to instantiate this attribute with an empty string.
- The *role* attribute specifies that the context of the identification assignment is 'document source'.
- The *items* attribute is a set of document representations or document files for which the document source information is valid.

ENTITY applied_external_identification_-assignment	Attribute Population	Remarks
assigned_id	type: label = STRING	recommended to be instantiated as ''
role	type: entity = IDENTIFICATION_ROLE	the role is required to have the name 'document source'
items	type: set of entity = product_definition (for document representations) or document_file	document objects for which the document source property applies

Pre-processor Recommendations: No standard mapping exists for the attribute applied_external_identification_assignment.assigned_id thus it is recommended to instantiate this attribute with an empty string. Applied_external_identification_assignment.role is required to be 'document source' in the given context.

Post-processor Recommendations: none specific

Related Entities: none specific

8.6.1.3 external_source

An external_source is the identification of a source of product related data. In the context of the document source property the external source provides the name of the source by which it can be accessed in a storage system.

Attributes

- The *source_id* attribute defines the source location of the referenced document objects.

ENTITY external_source	Attribute Population	Remarks
source_id	type source_item: STRING	specifies the location, where the object to which the identification assignment is applied, can be found.
description	type text: STRING	DERIVED: should not be instantiated

Pre-processor Recommendations: none specific

Post-processor Recommendations: none specific

Related Entities: none specific

8.7 Document type classification

The PDM Schema supports the assignment of document type information to document representations and document files. In the case where a document type property is assigned to the representation of a document the characteristics apply to all individual file objects, whereas in the case of an assignment to a document file the characteristics apply on an individual basis. The approach for this capability differs whether the document type of a document file or a document representation is modeled. The respective approaches are outlined in the following subsections.
The assignment of a classification system relative to which the classification has been done is not supported by the PDM Schema.

8.7.1 Document type classification for document files

A document_file provides in its attribute *kind* a pointer to the entity document_type. Document_type.product_data_type can be used for the classification of the type of the document_file.

8.7.1.1 document_type

Attributes

- The ***product_data_type*** attribute describes the type of product data represented by the document entity.

ENTITY document_type	Attribute Population	Remarks
product_data_type	type: identifier = string	

Pre-processor Recommendations: Where applicable the following values shall be used:
- 'geometry': The document file represents a shape model;
- 'NC data': The document file represents numerical control data;
- 'FE data': The document file represents finite element data;
- 'sample data': The document file represents measured data;
- 'process plan': The document file represents process planning data;
- 'check plan': The document file represents quality control planning data;
- 'drawing': The document file represents a technical drawing.

Post-processor Recommendations: none specific

Related Entities: This entity is required for each instance of the entity document.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#80 = DOCUMENT_FILE('file1', '', $, #90, '', '');
#90 = DOCUMENT_TYPE('geometry');
#100 = DOCUMENT_REPRESENTATION_TYPE('digital', #80);
```

Example 38: exchange file segment for document type classification related to files

8.7.2 Document type classification for representations of documents

In the use for document master identification, the attribute product_data_type of document_type is constrained by the entity document_product_equivalence to take one of the following values : 'configuration_controlled_document', 'configuration_controlled_document_version', or 'configuration_ controlled_document_definition'. In consequence document_type can not be used for the classification of document representations.

In AP214, the document type property information is instead captured in the description attribute of the related application_context. However, this mapping is the subject of several issues logged against this document. Refer to the Known Issues for this section (see 8.8.1).

8.8 Known Issues

8.8.1 Document type classification for managed documents

In AP214, the document type property information is captured in the description attribute of the related application_context. However, this mapping is the subject of several issues logged against this document. Attaching the document type property to the level of a document representation view definition is deemed to be not appropriate: the concept of a document type applies to all representation view definitions of all versions of the document.

8.8.1.1 description_attribute

The entity description_attribute provides a capability to assign additional attribute/value pairs to be described entities. In the given context a description_attribute entity is used to associate document type classification information to the application context that is linked to the document representation (i.e. the product_definition) via its product_definition_context (see diagram below).

Attributes

- The ***attribute_value*** attribute provides the description text.
- The *described_item* attribute points to the application_context for which additional description is provided via the description_attribute.

ENTITY description_attribute	Attribute Population	Remarks
attribute_value	type: label = text = STRING	
described_item	type: entity = application_context	the application context related to the document representation

Pre-processor Recommendations: Where applicable the following values for description_attribute.attribute_value shall be used in the given context:

- 'geometry': The document represents a shape model;
- 'NC data': The document represents numerical control data;
- 'FE data': The document represents finite element data;
- 'sample data': The document represents measured data;
- 'process plan': The document represents process planning data;
- 'check plan': The document represents quality control planning data;
- 'drawing': The document represents a technical drawing.

Post-processor Recommendations: none specific

Related Entities: none specific

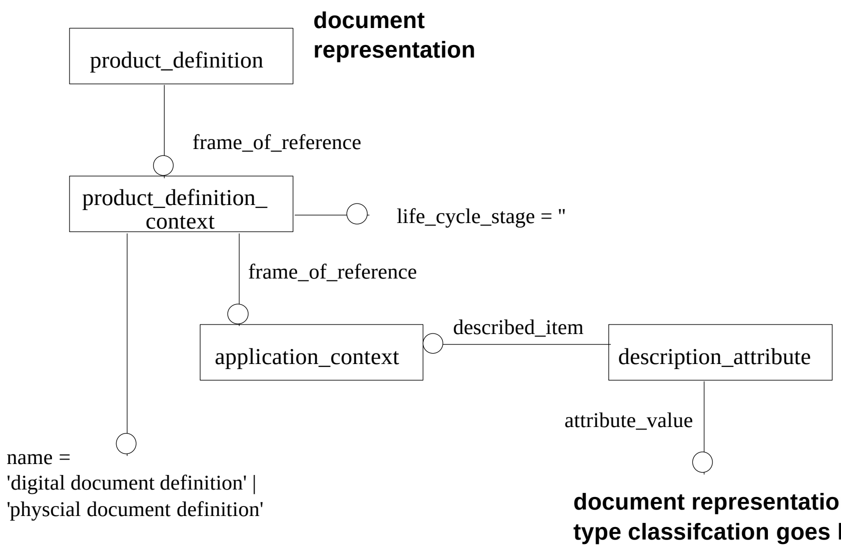


Diagram 44: Instantiation diagram for document type classification of document representation

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#50 = APPLICATION_CONTEXT('');
#70 = PRODUCT_DEFINITION_CONTEXT('digital document definition', #50, '');
#75 = DESCRIPTION_ATTRIBUTE('geometry',#75);
#110 = PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS('dig1',
   'digital document 1', #10, #70, (#80));
```

Example 39: exchange file segment for document type classification related to document representations

Complete instantiation example for document properties

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('Sample file for document properties'), '2;1');
FILE_NAME('', '15.06.1999, 10:39:20', ('N.N.'), (''), '', '', '');
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
DATA;
/* entities #10 to #70 represent a managed document with its context **/
```

```
#10 = PRODUCT_DEFINITION_FORMATION('1', 'version 1', #20);
#20 = PRODUCT('doc1', 'cad-model',
   'managed document representing a CAD model', (#30));
#30 = PRODUCT_CONTEXT('', #50, '');
#40 = PRODUCT_RELATED_PRODUCT_CATEGORY('document', '', (#20));
#50 = APPLICATION_CONTEXT('');
#60 = APPLICATION_PROTOCOL_DEFINITION('version 1.1', 'pdm_schema', 1999
   , #50);
#70 = PRODUCT_DEFINITION_CONTEXT('digital document definition', #50, '')
   ;

/* Entities #75 to #110 represent the digital document with id 'file1' */
/* that is associated to the document #20 via #110                     */
#75 = DESCRIPTION_ATTRIBUTE('geometry',#75);
#80 = DOCUMENT_FILE('file1', '', $, #90, '', '');
#90 = DOCUMENT_TYPE('geometry');
#100 = DOCUMENT_REPRESENTATION_TYPE('digital', #80);
#110 = PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS('dig1',
   'digital document 1', #10, #70, (#80));

/* Entities #120 to #130 represent an external hardcopy that is associated */
/* via the reference #125 to the part view defined by #550                */
#120 = DOCUMENT_FILE('hardcopy', '', $, #90, '', '');
#125 = APPLIED_DOCUMENT_REFERENCE(#120,''(#550));
#130 = DOCUMENT_REPRESENTATION_TYPE('physical', #120);

/* Entities #140 to #200 assign document format parameters to 'file1' (#80)
#140 = PROPERTY_DEFINITION('document property', '', #80);
#150 = PROPERTY_DEFINITION_REPRESENTATION(#140, #160);
#160 = REPRESENTATION('document format', (#170, #180, #190), #200);
#170 = DESCRIPTIVE_REPRESENTATION_ITEM('document format', 'VOXEL');
#180 = DESCRIPTIVE_REPRESENTATION_ITEM('size format', '5MB');
#190 = DESCRIPTIVE_REPRESENTATION_ITEM('character code', 'binary');
#200 = REPRESENTATION_CONTEXT('', 'document parameters');

/* Entities #210 to #250 assign document content parameters to 'file1' (#80)
#210 = PROPERTY_DEFINITION('document property', '', #80);
#220 = PROPERTY_DEFINITION_REPRESENTATION(#210, #230);
#230 = REPRESENTATION('document content', (#240, #250), #200);
#240 = DESCRIPTIVE_REPRESENTATION_ITEM('detail level', 'rough 3D shape');
#250 = DESCRIPTIVE_REPRESENTATION_ITEM('geometry type', 'solid model');

/* Entities #210 to #250 assign document creation parameters to 'file1' (#80)*/
#260 = PROPERTY_DEFINITION('document property', '', #80);
#270 = PROPERTY_DEFINITION_REPRESENTATION(#260, #280);
#280 = REPRESENTATION('document creation', (#290, #300, #310), #200);
#290 = DESCRIPTIVE_REPRESENTATION_ITEM('creating system', 'My CAD');
#300 = DESCRIPTIVE_REPRESENTATION_ITEM('operating system', 'Linux 2.1');
#310 = DESCRIPTIVE_REPRESENTATION_ITEM('creating interface',
   'export driver');

/* Entities #210 to #250 assign document format parameters to  */
/* the hardcopy document defined by #120                       */
#320 = PROPERTY_DEFINITION('document property', '', #120);
#330 = PROPERTY_DEFINITION_REPRESENTATION(#320, #340);
#340 = REPRESENTATION('document format', (#350, #360, #370), #200);
#360 = DESCRIPTIVE_REPRESENTATION_ITEM('size format', 'A0');

/* Entities #490 to #640 define the part master data and the association */
/* of the documents to the part                                          */
#490 = PRODUCT_CATEGORY_RELATIONSHIP('', $, #500, #510);
#500 = PRODUCT_RELATED_PRODUCT_CATEGORY('Part', '', (#520));
#510 = PRODUCT_RELATED_PRODUCT_CATEGORY('Detail', $, (#520));
```

```
#520 = PRODUCT('mp1', 'my_part', '', (#30));
#530 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', '', (#520));
#540 = PRODUCT_DEFINITION_FORMATION('1', '', #520);
#550 = PRODUCT_DEFINITION('ddid', 'design view on my_part', #540, #560);
#560 = PRODUCT_DEFINITION_CONTEXT('part definition', #50, '');
#590 = APPLIED_DOCUMENT_REFERENCE(#600, 'equivalence', (#550));
#600 = DOCUMENT('', '', '', #610);
#610 = DOCUMENT_TYPE('configuration controlled document version');
#620 = ROLE_ASSOCIATION(#630, #590);
#630 = OBJECT_ROLE('obligatory', '');
#640 = DOCUMENT_PRODUCT_EQUIVALENCE('equivalence', '', #600, #10);
ENDSEC;
END-ISO-10303-21;
```

9 Authorization

9.1 Organization and Person

The PDM Schema represents organizations and people in organizations as they perform functions related to other product data and data relationships. A person in the PDM Schema must exist in the context of some organization. An organization or a person in an organization is then associated with the data or data relationship in some role indicating the function being performed. Both people and organizations may have addresses associated with them. The address is entirely optional, it is done through the address entity being related to the person (through personal_address) or organization (through organizational_address).

9.1.1 Organization

The PDM Schema represents groups of people (e.g., companies, countries, etc.) through the organization entity. The identification or id data is optional. However, because this information is very important in providing unique identification to the organization or company, it is recommended that this field always be populated with unique data. If appropriate, a URL-like convention for the organization identifier may be useful, e.g., prostep.de. The name attribute should contain the common nomenclature of the organization. The description attribute may contain the full name of the organization or a textual explanation of its reason for existence.

The organization entity is related to certain constructs to identify the organizations responsible for them and how they are responsible. This relationship is defined through the applied_organization_assignment entity (which relates an organization in some role) to an entity. The role is established in the organization_role entity name attribute. The sections, which describe the use of the entity to which the organization is assigned, will identify the allowed values for the name attribute of the organization_role entity. The organization_role entity may have a description associated with it through the entity description_attribute and its attribute_value attribute.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements for organization are illustrated in Diagram 45.

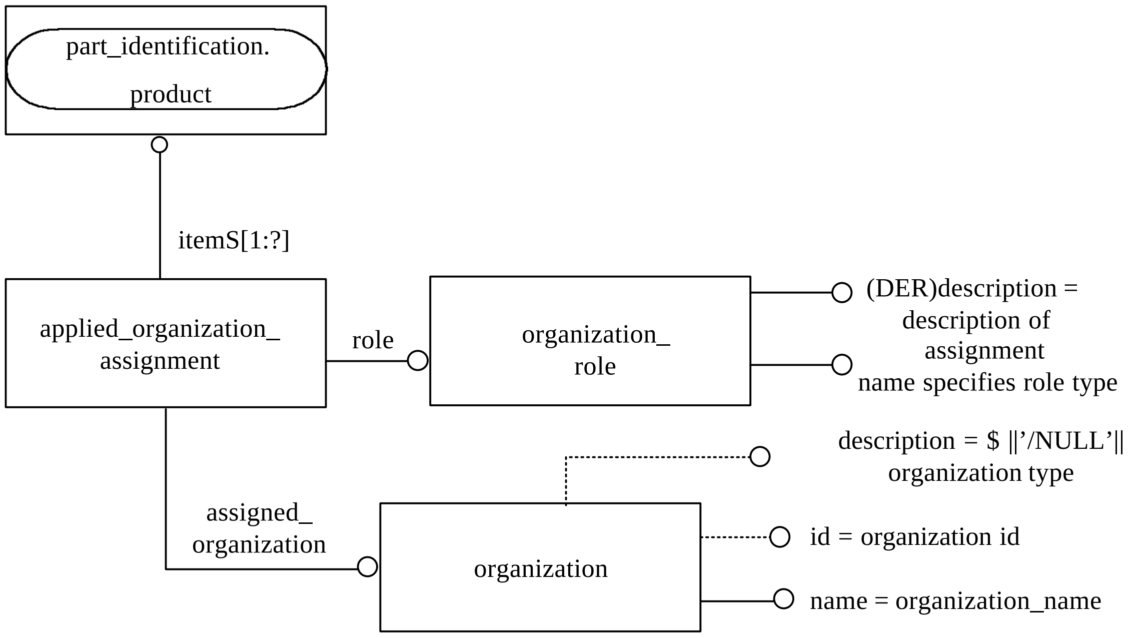


Diagram 45: Organization Instance Diagram

9.1.1.1 Organization

Entity represents the identification of an identified group of people gathered and structured for a purpose.

Attributes

- The ***id*** attribute is the identifier that distinguishes the organization.
- The ***name*** attribute is the label by which the organization is known.
- The ***description*** attribute characterizes the organization.

ENTITY organization	Attribute Population	Remarks
id	type: identifier = string	OPTIONAL it is recommended to instantiate this attribute
name	type: label = string	
description	type: text = string	OPTIONAL if available the type of organization shall be mapped into this attribute

Preprocessor Recommendations: All preprocessors should provide a unique organization id to eliminate ambiguities where organizations may have the same names. If the intended domain for the data is large, the reader is referred to ISO/IEC 8824-1, which can provide some guidance on creating unique identifiers. If appropriate, a URL like convention for the organization identifier may be used, e.g., prostep.de. A unique string obtained under ISO/IEC 8824-1 can be used as, or prefixed to, the organization identifier. For example, if the organization typically used an identifier of "93699" and the unique string were "USA", the unique value of the organization id would be "USA93699".

Postprocessor Recommendations: All postprocessors should make use of any provided information in the id attribute to eliminate ambiguities where organizations may have the same name.

Related Entities: There are no specific related entities.

9.1.1.2 applied_organization_assignment

This entity is a subtype of organization_assignment, allowing the representation of the assignment of an organization to some product data.

Attributes

- The ***assigned_organization*** is the organization which is to be associated with the product data.
- The ***role*** attribute specifies the purpose of the association of the organization_assignment with product data.
- The ***items*** attribute is a reference to the product data to which the organization is being assigned.

ENTITY applied_organization_assignment	Attribute Population	Remarks
assigned_organization	type: entity = organization	
role	type: entity = organization_role	
items	type: entity (defined by organization_item_select)	SET[1:?]

Preprocessor Recommendations: For applied_organization_assignments the potential organization_roles 'id owner' and 'creator' are recommended.

Postprocessor Recommendations: Postprocessors should support the assignment of the above roles to products, product_definition_formations and product_definitions (either related to parts or managed documents)

Related Entities: There are no specific related entities.

9.1.1.3 organization_role

This entity represents the role for the assignment of a person_and_organization to product data.

Attributes

- The ***name*** attribute is the label by which the organization_role is known.
- The ***description*** attribute characterizes the role with further descriptive text.

ENTITY organization_ role	Attribute Population	Remarks
name	type: label = string	
description	type: text = string	OPTIONAL

Preprocessor Recommendations: Role.name should be 'creator' or 'id owner' where applicable.

Postprocessor Recommendations: Postprocessors should at least support the above role names. Postprocessors should expect that some existing implementations may export files that use other role names such as 'design supplier' or 'designer'.

Related Entities*:* There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#310=APPLIED_ORGANIZATION_ASSIGNMENT(#320,#330,(#270));
#320=ORGANIZATION('ord_id_34','Doc Writers Com',$);
```

Example 40 : exchange file segment for organization

9.1.2 Person and Organization

The PDM Schema specifies information about people through the person entity. A person is identified by an id with other data representing their name and optionally titles which may apply to them. In populating the data, the id must be unique. This is typically not a problem when the person is taken in the context of some specific group such as a company or even country. In these instances, there are typically identifying numbers assigned to people. If the data being assembled is for worldwide consumption, the id must be unique in that domain.

In the PDM Schema, a person always exists in the context of some organization. The connection of people to organizations is accomplished through the person_and_organization entity. This entity is related to certain constructs to identify the people in organizations responsible for them and how they are responsible. This is specified through the applied_person_and_organization_assignment, which relates a person and organization in some role to an entity. The role is established in the name attribute of the person_and_organization_role entity. The person_and_organization_role and person_and_organization entity may have a description associated with it through the entity description_attribute and its attribute_value attribute. The person_and_organization entity may have a name associated with it through the entity name_attribute and its attribute_value attribute. This name describes the relationship of the person to the organization.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes used to support the requirements for person and organization are illustrated in Diagram 46.

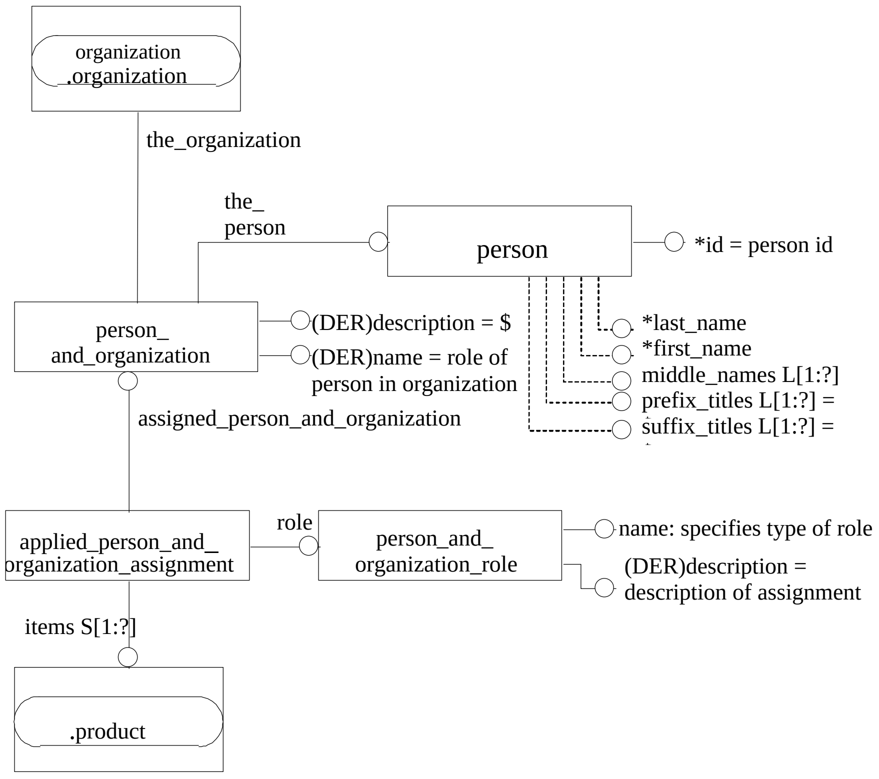


Diagram 46: Person and Organization Instance Diagram

9.1.2.1 person

This entity represents the identification of an individual. In the PDM Schema, a person always exists in the context of some organization.

Attributes

- The ***id*** attribute distinguishes the person.
- The ***last_name*** attribute has the last name of the person.
- The ***first_name*** attribute has the first name of the person.
- The ***middle_names*** attribute is a list of strings representing the middle names.
- The ***prefix_titles*** attribute is a list of text which specify the person's social and/or professional standing and appear before his or her names.
- The ***suffix_titles*** attribute is a list of text which specify the person's social and/or professional standing and appear before his or her names relevant for the person name that are used as a suffix.

ENTITY person	Attribute Population	Remarks
id	type: identifier = string	must be unique in the scope of the associated organization(s)
last_name	type: label = string	OPTIONAL it is recommended to instantiate this attribute
first_name	type: label = string	OPTIONAL
middle_names	type: label = string	OPTIONAL SET
prefix_titles	type: label = string	OPTIONAL SET
suffix_titles	type: label = string	OPTIONAL SET

Preprocessor Recommendations: All preprocessors should provide values for at least the last_name and first_name attributes for the person entity in order to provide a sense of meaning to the id attribute.
In cases where uniqueness of the id attribute may be a problem, preprocessors should prefix the id attribute with the organization id (as described in the following section) followed by a comma. For example, if the organization id value were "USA,93699" and the person id were "111111", the actual value of the person id would be "USA,93699,111111".

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.1.2.2 person_and_organization

This entity represents the identification of a person within an organization. In the PDM Schema, a person always exists in the context of some organization. The connection of people to organizations is accomplished through the person_and_organization entity.

Attributes

- The ***name*** attribute is a label which describes the role of the person in person_and_organization, i.e., the relationship between the person and the organization.
- The ***description*** attribute characterizes the person_and_organization.
- The ***the_person*** attribute is the person who is related to the organization.
- The ***the_organization*** attribute is the organization to which the person is related.

ENTITY person_and_organization	Attribute Population	Remarks
name		DERIVE the name is established via the entity name_attribute that has a label for the name as first attribute and references the person_and_organization in the second attribute
description		OPTIONAL
the_person	type: entity = person	
the_organization	type: entity = organization	

Preprocessor Recommendations: Description (i.e., the structure for the derivation of the attribute) shall not be instantiated. Pre-processors shall provide value 'member' for the *name* attribute supporting the general case of the role within an organization a person belongs to if no other values are available in the sending system.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.1.2.3 person_and_organization_role

This entity describes the assignment role for the assignment of a person_and_organization to product data.

Attributes

- The ***name*** attribute indicates the nature of the assignment.
- The ***description*** attribute characterizes the person_and_organization_role.

ENTITY person_and_organization_role	Attribute Population	Remarks
name	type: label = string	
description	type: text = string	DERIVED

Preprocessor Recommendations: See recommendations for organization_role.

Postprocessor Recommendations: See recommendations for organization_role.

Related Entities: There are no specific related entities.

9.1.2.4 applied_person_and_organization_assignment

This entity is a subtype of person_and_organization_assignment, and represents the assignment of a person_and_organization to product data.

REMARK: In the scope of this usage guide, applied_person_and_organization_assignment is used to associate information about responsibility, not to represent an electronic signature.

Attributes

- The ***assigned_person_and_organization*** attribute references the person_and_organization.
- The ***role*** attribute describes the nature of the assignment.
- The ***items*** attribute is a reference to the product data having the person_and_organization assigned.

ENTITY applied_person_and_ organization_assignment	Attribute Population	Remarks
assigned_person_and_organization	type: entity = person_and_organization	
role	type: entity = person_and_organization_role	
items	type: entity = person_and_organization_item select	SET[1:?]

Preprocessor Recommendations: This entity shall be used when the assignment involves an organization as well as a person related to that organization.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#120=APPLIED_PERSON_AND_ORGANIZATION_ASSIGNMENT(#130,#170,(#70));
#130=PERSON_AND_ORGANIZATION(#150,#140);
#140=ORGANIZATION('org_id','Miller Company',$);
#150=PERSON('p_id','Miller',$,$,$,$);
#155=NAME_ATTRIBUTE('employee',#130)
#310=APPLIED_ORGANIZATION_ASSIGNMENT(#320,#330,(#270));
#320=ORGANIZATION('ord_id_34','Doc Writers Com',$);
```

Example 41 : exchange file segment for person and organization

9.1.3 Address Assignment

The STEP PDM Schema provides for optional assignment of an address to both organizations and persons.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements for organizational address assignment are illustrated in Diagram 47. The mechanism for personal address assignment are similar.

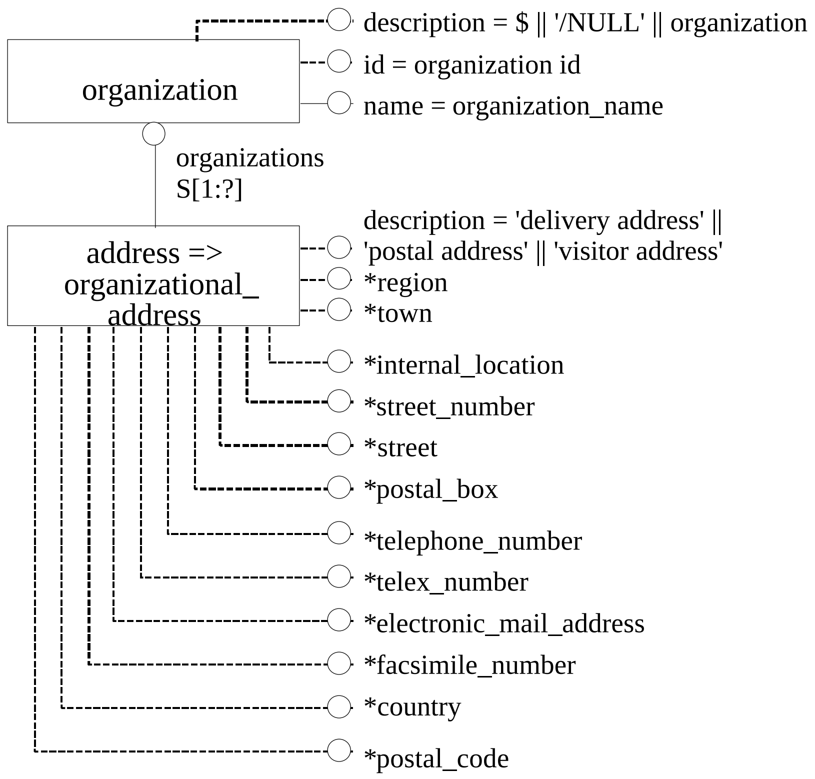


Diagram 47: Address Assignment Instance Diagram (for organization, similar for person)

9.1.3.1 address

This entity represents the identification of an address - a supertype entity is never instantiated as itself, but always as one of the defined subtypes organizational_address (see 9.1.3.2) or personal_address (see 9.1.3.3).

Attributes

- The ***internal_location*** attribute is an organization-defined address for internal mail delivery.
- The ***street_number*** attribute is the number of a location on a street.
- The ***street*** attribute is the name of the street.
- The ***postal_box*** attribute is the number of a postal box.
- The ***town*** attribute is the name of a town.
- The ***region*** attribute is the name of a region.
- The ***postal_code*** attribute is the code the is used by the country's postal service.
- The ***country*** attribute is the name of the country.
- The ***facsimile_number*** attribute is the number where facsimiles may be received.
- The ***electronic_mail_address*** attribute is the address where electronic mail messages may be received.
- The ***telephone_number*** is the number at which telephone calls may be received.
- The ***telex_number*** attribute is the number at which telex messages may be received.
- The ***description*** attribute characterizes the type of address.

ENTITY address	Attribute Population	Remarks
internal_location	type: label = string	OPTIONAL
street_number	type: label = string	OPTIONAL
street	type: label = string	OPTIONAL
postal_box	type: label = string	OPTIONAL
town	type: label = string	OPTIONAL
region	type: label = string	OPTIONAL
postal_code	type: label = string	OPTIONAL
country	type: label = string	OPTIONAL
facsimile_number	type: label = string	OPTIONAL
telephone_number	type: label = string	OPTIONAL
electronic_mail_address	type: label = string	OPTIONAL
telex_number	type: label = string	OPTIONAL
description	type: label = string	OPTIONAL

Preprocessor Recommendations: At least one of the optional attributes shall be instantiated.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.1.3.2 organizational_address

This entity is a subtype of address, represents the identification of address information relevant for an organization.

Attributes

- The ***description*** attribute characterizes the type address to distinguish between delivery, postal and visitor address.
- The ***organizations*** attribute is a reference to the organization at this address.

ENTITY organizational_address	Attribute Population	Remarks
internal_location		Same as supertype
street_number		Same as supertype
street		Same as supertype
postal_box		Same as supertype
town		Same as supertype
region		Same as supertype
postal_code		Same as supertype
country		Same as supertype
facsimile_number		Same as supertype
telephone_number		Same as supertype
electronic_mail_address		Same as supertype
telex_number		Same as supertype
description		Same as supertype
organizations		SET[1:?]

Preprocessor Recommendations: The description attribute shall be instantiated with either 'delivery address', 'postal address' or 'visitor address'. For each of these categories, at most one address should be instantiated related to organization. The delivery_address specifies the address where goods are delivered.
The postal address specifies the address where letter mail is delivered. A visitor address specifies the address where the organization receives visitors.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.1.3.3 personal_address

This entity is a subtype of address which represents the identification of address information to be applied to a person.

Attributes

- The ***people*** attribute lists the people that this location.

ENTITY personal_address	Attribute Population	Remarks
internal_location		Same as supertype
street_number		Same as supertype
street		Same as supertype
postal_box		Same as supertype
town		Same as supertype
region		Same as supertype
postal_code		Same as supertype
country		Same as supertype
facsimile_number		Same as supertype
telephone_number		Same as supertype
electronic_mail_address		Same as supertype
telex_number		Same as supertype
description		Same as supertype
people		SET[1:?]

Preprocessor Recommendations: At least one of the optional attributes shall be instantiated

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
#150=PERSON('p_id','Miller',$,$,$,$);
#155=NAME_ATTRIBUTE('employee',#130)
#160=PERSONAL_ADDRESS('office 321',$,$,$,$,$,$,$,'423 -21','423-
33','miller@my-com.com',$,$,'coordinates of Mr Miller');
#320=ORGANIZATION('ord_id_34','Doc Writers Com',$);
#340=ORGANIZATIONAL_ADDRESS($,'14a','E street','P.O 321','My
Town','Nice Region','postal code','My Country','345 - 21','345 -
31','info@company',$,$,$);
```

Example 42 : exchange file segment for address assignment

9.1.4 Recommendations for the assignment of person and organization

Person and organization data considered in this section is information about the responsibilities of persons and organizations in respect to product data. The entities for the assignments are person_and_organization_assignment and organization_assignment. The assignment entity has a role attribute that contains information about the meaning of the assignment. Of the variety of possible items to which the assignments of person and organizational data can be made, Table 1 gives the recommendations for scenarios and roles that should be supported.

	product (as part)	**product_-definition_-formation** (as part version)	**product_-definition** (as view on part version)	**product** (as document)	**product_-definition_-formation** (as document version)
person_-organization_assignment					
creator		**x**	**x**		**x**
id owner	**x**			**x**	

Table 1: Recommended Assignment of Person/Organization to Product Data

The interpretations of the roles are:

- the role 'creator' indicates that the referenced object has been created (i.e., originally defined) by the organization (or person_and_organization),
- the role 'id owner' describes the person or organization in which the id of the referenced object is valid.

In a given application scenario additional roles might be negotiated. Where applicable the following values may be used:

- 'custodian': The assigned Person or Organization is responsible for the continued existence and integrity of the referenced object;
- 'customer': The assigned Person or Organization acts as a purchaser or consumer of the referenced object;
- 'design supplier': The assigned Person or Organization is the one who is responsible for delivery or the data describing the referenced object;
- 'editor': The assigned Person or Organization is responsible for making any changes to any attribute of the referenced object;
- 'location': The assigned Organization is the place where the referenced object can be found or where it takes place;
- 'manufacturer': The assigned Person or Organization is the one who produces the actual (physical) object;
- 'owner': The assigned Person or Organization owns the referenced object, and has final say over its disposition and any changes to it;
- 'supplier': The assigned Person or Organization is the one who delivers the actual (physical) object (e.g., a dealer);
- 'wholesaler': The assigned Person or Organization is the one who is in the sales chain between the manufacturer and the supplier.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

A complete instantiation example for person and organization assignments is given in Example 43.

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('pdm_schema data', 'pdm_schema'), '2;1');
FILE_NAME('C:\document_xample.stp', '1999-5-20 T12:39:1', (''), (), '',
   '', '');
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
DATA;
/*   Entities #10 - #20 define part master data                    */
/*   To the product_definition #90 is a life cycle specific view on */
/*  that part to which organization and person are assigned in the  */
/*  role of the id owner                                            */
#10=APPLICATION_CONTEXT('');
#20=APPLICATION_PROTOCOL_DEFINITION('version
1.1','pdm_schema',1998,#10);
#30=PRODUCT_DEFINITION_CONTEXT_ROLE('',$);
#40=APPLICATION_CONTEXT('mechanical design');
#45=APPLICATION_PROTOCOL_DEFINITION('version
1.1','pdm_schema',1999,#40);
#50=PRODUCT_DEFINITION_CONTEXT('part definition',#40,'design');
#60=PRODUCT_CONTEXT('',#10,'');
#70=PRODUCT('MP-03-2','my part',$,(#60));
#80=PRODUCT_DEFINITION_FORMATION('03','3rd modification',#70);
#90=PRODUCT_DEFINITION('/NULL',$,#80,#50);
#100=PRODUCT_DEFINITION_CONTEXT_ASSOCIATION(#90,#50,#30);
#110=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#70));

/* Entities #120 to #170 model the assignment of a person within an */
/* organization to the part. The role of this person/organization  */
/* is that of 'id owner'                                      */
#120=APPLIED_PERSON_AND_ORGANIZATION_ASSIGNMENT(#130,#170,(#70));
#130=PERSON_AND_ORGANIZATION(#150,#140);
#140=ORGANIZATION('org_id','Miller Company',$);
#150=PERSON('p_id','Miller',$,$,$,$);
#155=NAME_ATTRIBUTE('employee',#130)
#160=PERSONAL_ADDRESS('office 321',$,$,$,$,$,$,$,'423 -21','423-
33','miller@my-com.com',$,$,'coordinates of Mr Miller');
#170=PERSON_AND_ORGANIZATION_ROLE('id owner');

/*   Entities #240 to #300 model a managed document (document as
product)  */
/*   The document gets an assigned organization in the role of
'creation'  */

#240=APPLICATION_CONTEXT('');
#245=APPLICATION_PROTOCOL_DEFINITION('version
1.1','pdm_schema',1999,#240);
#250=PRODUCT_DEFINITION_CONTEXT('digital document definition',#240,'');
#260=PRODUCT_CONTEXT('',#240,'');
#270=PRODUCT('doc_4711','packaging guide','packaging guideline for
part',(#260));
#280=PRODUCT_DEFINITION_FORMATION('ver3.1','version 3.1',#270);
#290=PRODUCT_DEFINITION('id of digital document','digital document for
representing guideline ver3.1',#280,#250);
#300=PRODUCT_RELATED_PRODUCT_CATEGORY('document',$,(#270));

/* Entities #310 to #340 model the assignment of an organization   */
/* to the document defined via #270. The info transferred with the */
/* assignment is that the organization has created the document    */
#310=APPLIED_ORGANIZATION_ASSIGNMENT(#320,#330,(#270));
#320=ORGANIZATION('ord_id_34','Doc Writers Com',$);
```

```
#330=ORGANIZATION_ROLE('creation');
#340=ORGANIZATIONAL_ADDRESS($,'14a','E street','P.O 321','My
Town','Nice Region','postal code','My Country','345 - 21','345 -
31','info@company',$,$,$);
ENDSEC;
END-ISO-10303-21;
```

Example 43: exchange file for complete person and organization

9.1.5 Known issues

9.1.5.1 Organization type

In the AP214 ARM, organization has a mandatory type attribute that maps into organization.description. Organization.description is an optional attribute in the PDM Schema. In the PDM Schema, it is not required that organization.description be instantiated.

9.2 Approval

Approving in the PDM Schema is accomplished by establishing an approval entity and relating it to some construct through an applied_approval_assignment entity. The applied_approval_assignment entity may have a role associated with it through the entity role_association and its related object_role entity. It is recommended that the approval entity have associated approval_person_organization and approval_date_time constructs to specify the approver (or multiple approvers) and the date when the current approval status was assigned.

In a number of STEP APs, constructs that require an approval are allowed only one approval assignment. This might lead to the misconception that only one person at one date/time can approve something. This is not the case. The approval constructs actually support an approval cycle. This cycle may need multiple signatures. This explains the need for the approval_status of 'not yet approved' - until all signatures are available to switch this status to 'approved'.

However, an approval may only need one signature. If there is only one approval_person_organization and the approval_status is 'approved', the approval_date_time indicates that this person and/or organization approved it on this date and/or time.

The approval_date_time always records the date/time that the status of the approval was last changed. Year values for dates should always be expressed with four digits, e.g., 1999. The approval_date_time entity may have a role associated it through the entity role_association and its related object_role entity. It does not necessarily record when the approval was given by the approval_person_organization, as there can be multiple approval_person_organizations related to an approval entity.

When an approval event is a cycle which requires multiple people to concur on possibly differing dates and/or times, multiple approval_person_organization constructs may be associated with the approval. The approval_person_organization entity has a role associated with it by the entity approval_role. If no more specific information is available, the role 'approver' is recommended. The multiple dates/times are recorded through the relation of an applied_date_and_time_assignment entity (date_and_time_assignment to approval_person_organization) with the associated date_time_role of "sign off ". In the multiple person/organization cycle case, as in the case of a single sign-off, the approval_date_time indicates when the status of the approval was last changed.

The Instance Model: EXPRESS entities and attributes

The EXPRESS entities and attributes essential to support the requirements for approval are illustrated in Diagram 48.

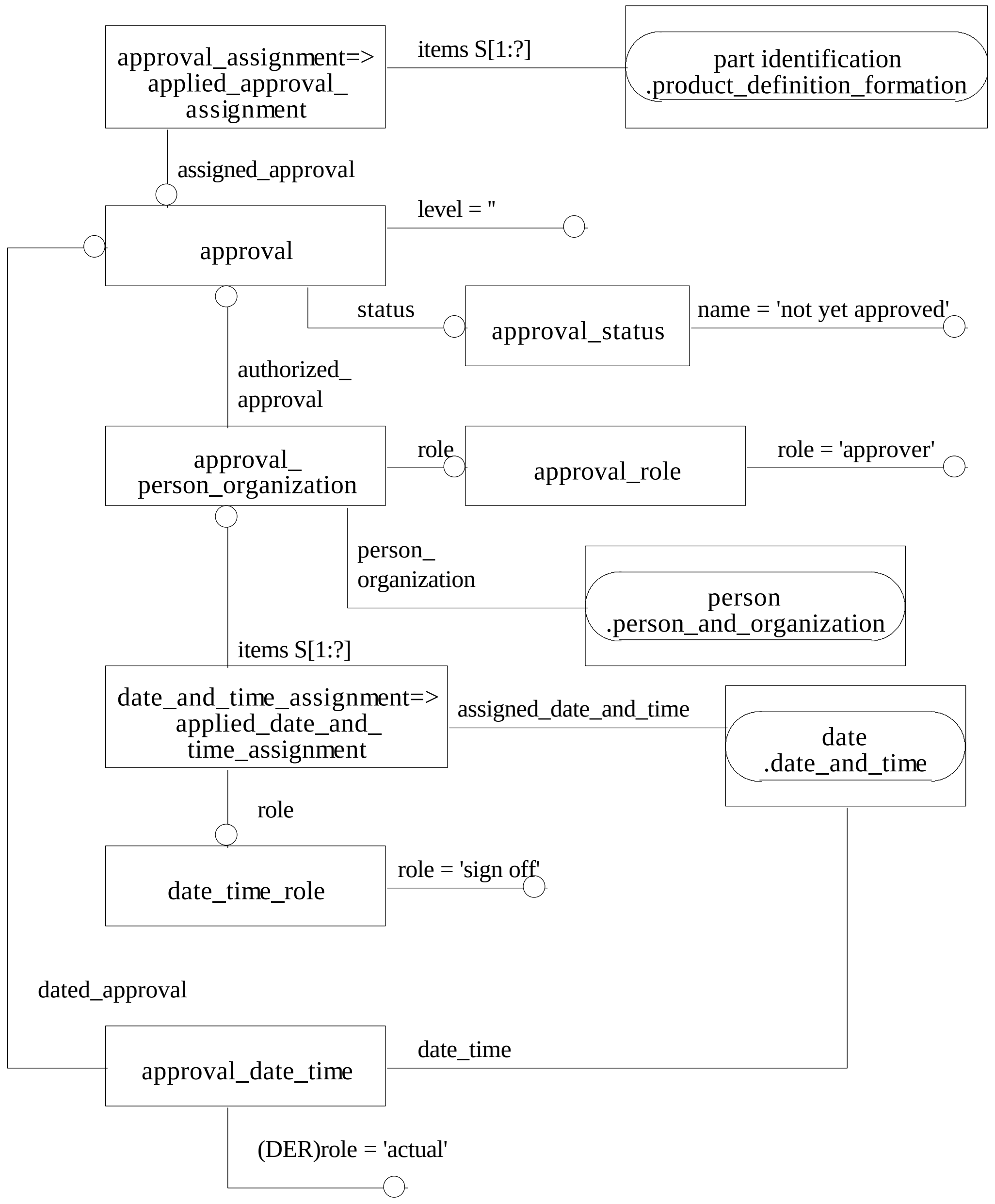


Diagram 48: Approval Instance Diagram

9.2.1.1 approval

This entity describes the state of acceptance of some product data.

Attributes
- The ***status*** attribute specifies the judgement made about the product data that is the subject of the approval.
- The ***level*** attribute describes the type or the level of approval in terms of its usage.

ENTITY approval	Attribute Population	Remarks
status	type: entity = approval_status	
level	type: label	

Preprocessor Recommendations: There is no standard mapping for the approval level attribute. Since there is no standard mapping in the PDM application domain for this attribute, it is recommended that this attribute contains an empty string as minimal content or any appropriate or mutually agreed upon string.

Postprocessor Recommendations: Since there is no standard mapping for the approval level attribute, postprocessors should not assign any processing significance to this value.

Related Entities: There are no specific related entities.

9.2.1.2 approval_status

This entity represents a statement made by technical personnel or management personnel whether certain requirements are met.

Attributes
- The ***name*** attribute describes the terms characterizing the approval_status.

ENTITY approval_status	Attribute Population	Remarks
name	type: label	

Preprocessor Recommendations: The approval_status name attribute is recommended to only be "approved", "not yet approved", "disapproved" or "withdrawn".

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.2.1.3 approval_assignment

This entity represents the assignment of an approval to some product data. It is an abstract supertype entity never instantiated as itself, always as the subtype applied_approval_assignment.

Attributes
- The ***assigned_approval*** attribute references the approval that is being assigned.
- The ***role*** attribute describes the nature of the assignment.

ENTITY approval_assignment	Attribute Population	Remarks
assigned_approval	type: entity = approval	
role	type: entity = object_role	DERIVE

Preprocessor Recommendations: should be instantiated only as the subtype applied_approval_assignment.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.2.1.4 applied_approval_assignment

This entity is a subtype of approval_assignment, allowing the representation of the assignment of an approval to some product data.

Attributes

- The ***items*** attribute is a reference to the product data to which the approval is being assigned.

ENTITY applied_approval_assignment	Attribute Population	Remarks
assigned_approval		Same as supertype
role		Same as supertype
items		SET[1:?]

Preprocessor Recommendations: All preprocessors should use non-defaulted data or user-input data for the values assigned for the approvers and approval date for product_definition_formation entities. The approvers and approval dates can be extrapolated from the version creator and approval data if other appropriate data is unavailable.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.2.1.5 approval_person_organization

This entity specifies who is responsible for the approval.

Attributes

- The ***person_organization*** attribute is a reference to the person or person_and_organization who made the approval.
- The ***authorized_approval*** attribute is a reference to the approval effected by the person and/or organization.
- The ***role*** attribute describes the nature of the association.

ENTITY approval_person_organization	Attribute Population	Remarks
person_organization	type: SELECT	
authorized_approval	type: entity = approval	
role	type: entity = approval_role	

Preprocessor Recommendations: It is recommended that all approval_person_organization instances have associated applied_date_and_time_assignment entities to provide complete clarity.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.2.1.6 approval_role

This entity represents the identification of the assignment role for an approval_person_organization.

Attributes

- The ***role*** attribute describes the terms characterizing the approval_role.
- The ***description*** attribute specifies the word or group of words used to refer to the approval_role.

ENTITY approval_role	Attribute Population	Remarks
role	type: label	
description	type: text	DERIVE

The approval_role entity may have a description associated with it through the entity description_attribute and its attribute_value attribute.

Preprocessor Recommendations: If no appropriate data for the approval_role role attribute (why this person and organization is approving) is available it is recommended that this attribute contain the value "approver".

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

9.2.1.7 approval_date_time

This entity specifies when the approval was given. Year values for dates should always be expressed with four digits, e.g., 1999.

Attributes

- The ***date_time*** attribute is a reference to the date and/or time when the approval was made.
- The ***dated_approval*** attribute is a reference to the approval effected by the date and/or time.
- The ***role*** attribute describes the nature of the association.

ENTITY approval_date_time	Attribute Population	Remarks
date_time	type: SELECT	
dated_approval	type: entity = approval	
role	type: entity = object_role	DERIVE

Preprocessor Recommendations: The approval_date_time.role attribute may contain the value 'actual' to indicate this date/time is the actual (not planned) date that the approval_status related to the dated_approval was assigned.

Postprocessor Recommendations: There are no specific postprocessor recommendations.

Related Entities: There are no specific related entities.

The Instance Model: STEP exchange file format (ISO10303 Part 21 syntax)

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('pdm_schema data', 'pdm_schema'), '2;1');
FILE_NAME('C:\approval_example.stp', '1999-5-20 T12:39:1', (''), (),
'',
   '', '');
FILE_SCHEMA(('PDM_SCHEMA {1.1}'));
ENDSEC;
DATA;

/* part version to which an approval is applied */
#10 = PRODUCT_DEFINITION_FORMATION('02', 'lever modified', #20);
#20 = PRODUCT('K01-42051', 'Bicycle Bell RX25B', '', (#30));
#30 = PRODUCT_CONTEXT('', #40, '');
#40 = APPLICATION_CONTEXT('mechanical design');
```

```
#50 = PRODUCT_RELATED_PRODUCT_CATEGORY('part', '', (#20));
#60=APPLICATION_PROTOCOL_DEFINITION('version1.1','pdm_schema',1998,#40;

/* information on person and organization */
#280 = ORGANIZATION('O-0709', 'Logus GmbH', 'company');
#330 = PERSON_AND_ORGANIZATION(#340, #280);
#340 = PERSON('P-0661', 'Rose', 'Peter', $, $, $);
#350 = NAME_ATTRIBUTE('employee', #330);

/* approval definition / association to product_definition_formation */
#870 = APPROVAL_STATUS('not yet approved');
#880 = APPROVAL(#870, '');
#890 = APPLIED_APPROVAL_ASSIGNMENT(#880, (#10));

/* definition of digital signature */
#900 = APPROVAL_PERSON_ORGANIZATION(#330, #880, #910);
#910 = APPROVAL_ROLE('approver');
#920 = APPLIED_DATE_AND_TIME_ASSIGNMENT(#970, #930, (#900));
#930 = DATE_TIME_ROLE('sign off');

/* definition of actual approval date */
#940 = APPROVAL_DATE_TIME(#970,#880);
#950 = ROLE_ASSOCIATION(#960, #940);
#960 = OBJECT_ROLE('actual', $);

/* definition of date and time */
#970 = DATE_AND_TIME(#980, #990);
#980 = CALENDAR_DATE(1999, 14, 5);
#990 = LOCAL_TIME(0, 0, 0.0E+000, #1000);
#1000 = COORDINATED_UNIVERSAL_TIME_OFFSET(1, $, .AHEAD.);
ENDSEC;
END-ISO-10303-21;
```

Example 44: exchange file segment for approval

9.2.2 Known Issues

9.2.2.1 Sign off vs. actual to describe the role of a date/time assignment for a digital signature

The AP214 ARM says that the role name ‘actual’ shall be used for a date_time_role specifying the assignment of date/time to approval_person_organization. This role name is already used for approval_date_time to specify the actual date of an approval. In AP203 it is recommended to use ‘sign off’ as role name in date_time_role. For these reasons, this usage guide recommends use of that value.
A ballot comment has to be issued against AP214 DIS to reflect the agreement.

9.2.2.2 Scope in which an approval is valid

Some application protocols provide the capability to specify the scope in which a given approval is valid. A way to support such functionality is to use an applied_organization_assignment that relates an organization – in the role 'scope' – to an approval. This assignment is not supported by the PDM Schema 1.1

Annex A PDM Schema EXPRESS listing

A.1 PDM Schema short form

http://www.pdm-if.org/pdm_schema/pdm11_sf.exp

A.2 PDM Schema long form

http://www.pdm-if.org/pdm_schema/pdm11lf.exp

Annex B PDM Schema EXPRESS-G Diagrams

http://www.pdm-if.org/pdm_schema/pdm11-exg.pdf

Annex C Document Change Log

Changes to Release 1.0

- Removal of creation information recommendation in section 1.1.1,
- Removal of creation information recommendation in 1.1.2,
- Addition of Document Identification (section 4),
- Addition of File Identification (section 5),
- Addition of Authorization - Person and Organization (section 9.1),
- Addition of Authorization - Approval (section 9.2).

Changes to Release 2.0

- See Closed Issues in the associated PDM Schema Usage Guide Issue Table

Changes to Release 3.0

- Addition of section on part properties (section 2)
- Addition of section on part structure and relationships (section 3)
- Addition of section on document properties (section 8)